

Completeness for Two Left-Sequential Logics

MSc Thesis (*Afstudeerscriptie*)

written by

D.J.C. Staudt

(born January 12, 1987 in Amsterdam, The Netherlands)

under the supervision of **Dr. Alban Ponse**, and submitted to the Board of
Examiners in partial fulfillment of the requirements for the degree of

MSc in Logic

at the *Universiteit van Amsterdam*.

Date of the public defense:
May 31st, 2012

Members of the Thesis Committee:
Dr. Alexandru Baltag
Dr. Inge Bethke
Dr. Alban Ponse
Prof. Dr. Frank Veltman



INSTITUTE FOR LOGIC, LANGUAGE AND COMPUTATION

Abstract

Left-sequential logics provide a means for reasoning about (closed) propositional terms with atomic propositions that may have side effects and that are evaluated sequentially from left to right. Such propositional terms are commonly used in programming languages to direct the flow of a program. In this thesis we explore two such left-sequential logics. First we discuss Fully Evaluated Left-Sequential Logic, which employs a full evaluation strategy, i.e., to evaluate a term every one of its atomic propositions is evaluated causing its possible side effects to occur. We then turn to Short-Circuit (Left-Sequential) Logic as presented in [BP10b], where the evaluation may be ‘short-circuited’, thus preventing some, if not all, of the atomic propositions in a term being evaluated. We propose evaluation trees as a natural semantics for both logics and provide axiomatizations for the least identifying variant of each. From this, we define a logic with connectives that prescribe a full evaluation strategy as well as connectives that prescribe a short-circuit evaluation strategy.

Contents

1	Introduction	7
2	Free Fully Evaluated Logic (FFEL)	9
2.1	FEL Normal Form	14
2.2	Tree Structure	17
2.3	Completeness	20
3	Free Short-Circuit Logic (FSCL)	23
3.1	SCL Normal Form	27
3.2	Tree Structure	31
3.3	Completeness	34
4	Relating FFEL to FSCL and Proposition Algebra	37
4.1	Relating FFEL to Proposition Algebra	38
4.2	EqFSCL axiomatizes FSCL	39
4.3	FFEL is a sublogic of FSCL	40
5	Conclusion and Outlook	43
A	Proofs for FFEL	47
A.1	Correctness of f	47
A.2	Correctness of g	51
B	Proofs for FSCL	55
B.1	Correctness of f	55
B.2	Correctness of g	60
	Bibliography	63

Introduction

In computer programming it is common to use propositional terms to control the flow of a program. These expressions occur for example in `if` and `while` statements. Although at first sight it might appear as though these expressions are terms of a Boolean algebra, it turns out that their semantics are not governed by ordinary Propositional Logic (PL). The reason is that many programming languages allow arbitrary instructions, e.g., method calls, to occur as atomic propositions in such terms. Those instructions may have side effects. Therefore the truth value of a term may depend on the state of the execution environment, e.g., the operating system or the Java Virtual Machine. This state in turn can also be altered by the evaluation of (part of) a term. For example, in the term `x && y`, a side effect of the evaluation of `x` may be that any subsequent evaluation of `y` yields false. In that case `x && y` will yield false, while `y && x` may yield true, i.e., conjunction is no longer commutative.

This shows that, unlike in PL, the evaluation strategy that is used impacts the truth values of terms. Most programming languages evaluate such terms sequentially from left to right. We refer to such an evaluation strategy as a *left-sequential evaluation strategy*. In addition, some programming languages offer connectives that are evaluated using a *short-circuit (left-sequential) evaluation strategy*, such as `&&` and `||` in the Java programming language, see, e.g., [AGH06]. A short-circuit evaluation strategy is one that evaluates only as much of a propositional term as is necessary to determine its truth value. For example, when evaluating the term `x && y`, if `x` evaluates to false, the entire term will be false, regardless of the truth value of `y`. In that case the evaluation is ‘short-circuited’ and `y` is never evaluated. An evaluation strategy that always evaluates terms in their entirety is called a *full (left-sequential) evaluation strategy*. In Java, for example, the connectives `&` and `|` are evaluated using a full evaluation strategy. Some languages provide both short-circuited and full versions of the binary connectives, as Java does, thus allowing the programmer to write terms that prescribe a mixed evaluation strategy.

In [BP11], Bergstra and Ponse introduce Proposition Algebra as a means for reasoning about sequential evaluations of propositional terms using a ternary conditional connective, $y \triangleleft x \triangleright z$, to be read as ‘if x then y else z ’. In [BP10b], they define *Short-Circuit Logic* (SCL) in terms of Proposition Algebra using left-sequential versions of the standard logical connectives. SCL formalizes equality between propositional terms that are evaluated with a short-circuit evaluation strategy. They use \neg for negation, $\overset{\circ}{\vee}$ for (short-circuit) left-sequential disjunction and $\overset{\circ}{\wedge}$ for (short-circuit) left-sequential conjunction. The position of the circle indicates the direction of the evaluation, i.e., from left to right. The negation symbol does not have a circle, because it has only one possible evaluation strategy, i.e., evaluate the negated subterm and then negate the

result. Several variants of SCL are described in [BP10b], ranging from the least identifying, Free SCL (FSCL), to the most identifying, Static SCL, which corresponds to PL. The only difference between Static SCL and PL is that the connectives in Static SCL are left-sequential and that the evaluation is short-circuited. Several semantics have been given for SCL, such as valuation congruences [BP11], Hoare-McCarthy algebras [BP10a] and truth tables [Pon11].

In [Blo11] Blok first defined *Fully Evaluated Left-Sequential Logic*, or *Fully Evaluated Logic* (FEL) for short. FEL is used for dealing with terms that are to be evaluated using a full evaluation strategy. Blok refers to this logic as Side-Effecting Logic, but we prefer the name FEL so that we do not implicitly discount SCL as a logic that can be used for reasoning about side effects. To allow for a mixed setting of FEL and SCL, we must distinguish the symbols used in FEL-terms from those of Bergstra and Ponse. We use \blacktriangleleft for full left-sequential conjunction and \blacktriangleright for full left-sequential disjunction. We still use \neg for negation, because it is evaluated with the same strategy as in SCL. We can now view the open circles in \triangleleft and \triangleright as indicating short-circuiting while the closed circles of \blacktriangleleft and \blacktriangleright indicate full evaluations. No variants of FEL other than Free FEL (FFEL) have yet been formally defined.

In this thesis we will also define a logic for reasoning about propositional terms that contain both short-circuit left-sequential connectives and full left-sequential connectives. We refer to a logic that offers both types of connectives as a *general left-sequential logic*.

The main differences between these left-sequential logics and PL is that they employ a left-sequential evaluation strategy and that their atoms may have side effects. We note that logics employing right-sequential evaluation strategies can easily be expressed in terms of their left-sequential counterparts. We study the left-sequential versions because most programming languages are oriented left-to-right, mainly due to having been developed in the Western world. Although side effects are well understood in programming, see e.g., [BW96] or [Nor97], they are often explained without a general formal definition. In Chapter 5 we will substantiate our claim that these logics can be used to formally reason about propositional terms whose atoms may have side effects. We note that both SCL and FEL are sublogics of PL, in the sense that they identify fewer propositions, i.e., closed terms, although both have extreme variants that are equivalent with PL.

We start in Chapter 2 by formally introducing FFEL and the set of equations EqFFEL. We prove that EqFFEL is an axiomatization of FFEL. In Chapter 3 we introduce FSCL_{SE} as an alternative semantics for the Proposition Algebra semantics of FSCL. We also discuss the set of equations EqFSCL and prove that it axiomatizes FSCL_{SE}. In Section 4.2 we prove that it also axiomatizes FSCL. Chapters 2 and 3 are written to be self-contained, hence there is some duplication of definitions and narrative. In Chapter 4 we investigate the relations between FFEL and FSCL. We show in Section 4.1 that FFEL can also be expressed in terms of Proposition Algebra. In Section 4.2 we prove that FSCL_{SE} is equivalent to FSCL. In Section 4.3 we show that FFEL is a sublogic of FSCL and use this fact to define a general left-sequential logic. We conclude with some final remarks and provide an outlook for further study in Chapter 5.

CHAPTER 2

Free Fully Evaluated Logic (FFEL)

In this chapter we define Free Fully Evaluated Left-Sequential Logic, or Free Fully Evaluated Logic (FFEL) for short, and the set of equations EqFFEL, which we will prove axiomatizes FFEL in Section 2.3. We start by defining FEL-terms, which are built up from atomic propositions, referred to as atoms, the truth value constants **T** for true and **F** for false and the connectives \neg for negation, \blacktriangleleft for full left-sequential conjunction and \blacktriangleright for full left-sequential disjunction.

Definition 2.1. *Let A be a countable set of atoms. **FEL-terms** (**FT**) have the following grammar presented in Backus-Naur Form.*

$$P \in \text{FT} ::= a \in A \mid \mathbf{T} \mid \mathbf{F} \mid \neg P \mid (P \blacktriangleleft P) \mid (P \blacktriangleright P)$$

If $A = \emptyset$ then the resulting logic is trivial.

Let us return for a moment to our motivation for left-sequential logics, i.e., propositional terms as used in programming languages. We will consider the FEL-term $a \blacktriangleright b$ and informally describe its evaluation, naturally using a full evaluation strategy. We start by evaluating a and let its yield determine our next action. If a yielded **F** we proceed by evaluating b , i.e., the yield of the term as a whole will be the yield of b . If a yielded **T**, we already know at this point that $a \blacktriangleright b$ will yield **T**. We still evaluate b though, but ignore its yield and instead have the term yield **T**. Evaluating a subterm even though its yield is not needed to determine the yield of the term as a whole is the quintessence of a full evaluation strategy.

Considering the more complex term $(a \blacktriangleright b) \blacktriangleleft c$, we find that we start by evaluating $a \blacktriangleright b$ and if it yielded **T** we proceed by evaluating c . If it yielded **F** we still evaluate c , even though we know that the term as a whole will now yield **F**. The discussion of the evaluations of these terms may have evoked images of trees in the mind of the reader. We will indeed define equality of FEL-terms using (evaluation) trees. We define the set \mathcal{T} of binary trees over A with leaves in $\{\mathbf{T}, \mathbf{F}\}$ recursively. We have that

$$\mathbf{T} \in \mathcal{T}, \quad \mathbf{F} \in \mathcal{T}, \quad \text{and} \quad (X \trianglelefteq a \trianglerighteq Y) \in \mathcal{T} \text{ for any } X, Y \in \mathcal{T} \text{ and } a \in A.$$

In the expression $X \trianglelefteq a \trianglerighteq Y$ the root is represented by a , the left branch by X and the right branch by Y . As is common, the depth of a tree X is defined recursively by $d(\mathbf{T}) = d(\mathbf{F}) = 0$ and for all $a \in A$, $d(Y \trianglelefteq a \trianglerighteq Z) = 1 + \max(d(Y), d(Z))$. Our reason for choosing this particular

notation for trees, out of the many that exist, is explained in Chapter 4. We shall refer to trees in \mathcal{T} as evaluation trees, or simply trees for short. Figure 2.1 shows the trees corresponding to the evaluations of $(a \blacktriangledown b) \blacktriangleleft c$ and $(a \blacktriangleleft b) \blacktriangledown c$.

Returning to our example, we have seen that the tree corresponding to the evaluation of $(a \blacktriangledown b) \blacktriangleleft c$ can be composed from the tree corresponding to the evaluation of $a \blacktriangledown b$ and that corresponding to the evaluation of c . We said above that if $a \blacktriangledown b$ yielded T , we would proceed with the evaluation of c . This can be seen as replacing each T -leaf in the tree corresponding to the evaluation of $a \blacktriangledown b$ with the tree that corresponds to the evaluation of c . Formally we define the leaf replacement operator, ‘replacement’ for short, on trees in \mathcal{T} as follows. Let $X, X', X'', Y, Z \in \mathcal{T}$ and $a \in A$. The replacement of T with Y and F with Z in X , denoted $X[\mathsf{T} \mapsto Y, \mathsf{F} \mapsto Z]$, is defined recursively as

$$\begin{aligned} \mathsf{T}[\mathsf{T} \mapsto Y, \mathsf{F} \mapsto Z] &= Y \\ \mathsf{F}[\mathsf{T} \mapsto Y, \mathsf{F} \mapsto Z] &= Z \\ (X' \trianglelefteq a \trianglerighteq X'')[\mathsf{T} \mapsto Y, \mathsf{F} \mapsto Z] &= X'[\mathsf{T} \mapsto Y, \mathsf{F} \mapsto Z] \trianglelefteq a \trianglerighteq X''[\mathsf{T} \mapsto Y, \mathsf{F} \mapsto Z]. \end{aligned}$$

We note that the order in which the replacements of the leaves of X is listed inside the brackets is irrelevant. We will adopt the convention of not listing any identities inside the brackets, i.e.,

$$X[\mathsf{F} \mapsto Y] = X[\mathsf{T} \mapsto \mathsf{T}, \mathsf{F} \mapsto Y].$$

Furthermore we let replacements associate to the left. We also use that fact that

$$X[\mathsf{T} \mapsto Y][\mathsf{F} \mapsto Z] = X[\mathsf{T} \mapsto Y, \mathsf{F} \mapsto Z]$$

if Y does not contain F , which can be shown by a trivial induction. Similarly,

$$X[\mathsf{F} \mapsto Z][\mathsf{T} \mapsto Y] = X[\mathsf{T} \mapsto Y, \mathsf{F} \mapsto Z]$$

if Z does not contain T . We now have the terminology and notation to formally define the evaluation of FEL-terms.

Definition 2.2. Let A be a countable set of atoms and let \mathcal{T} be the set of all finite binary trees over A with leaves in $\{\mathsf{T}, \mathsf{F}\}$. We define the unary **Full Evaluation** function $\mathsf{FE} : \mathsf{FT} \rightarrow \mathcal{T}$ as:

$$\begin{aligned} \mathsf{FE}(\mathsf{T}) &= \mathsf{T} \\ \mathsf{FE}(\mathsf{F}) &= \mathsf{F} \\ \mathsf{FE}(a) &= \mathsf{T} \trianglelefteq a \trianglerighteq \mathsf{F} && \text{for } a \in A \\ \mathsf{FE}(\neg P) &= \mathsf{FE}(P)[\mathsf{T} \mapsto \mathsf{F}, \mathsf{F} \mapsto \mathsf{T}] \\ \mathsf{FE}(P \blacktriangleleft Q) &= \mathsf{FE}(P)[\mathsf{T} \mapsto \mathsf{FE}(Q), \mathsf{F} \mapsto \mathsf{FE}(Q)[\mathsf{T} \mapsto \mathsf{F}]] \\ \mathsf{FE}(P \blacktriangledown Q) &= \mathsf{FE}(P)[\mathsf{T} \mapsto \mathsf{FE}(Q)[\mathsf{F} \mapsto \mathsf{T}], \mathsf{F} \mapsto \mathsf{FE}(Q)]. \end{aligned}$$

Note that because we require A to be a set, \mathcal{T} is also a set. By a trivial induction we can show that all trees in the image of FE are perfect binary trees, i.e., all their paths are of equal length. As we can see from the definition on atoms, the evaluation continues in the left branch if an atom yields T and in the right branch if it yields F . Revisiting our example once more, we indeed see how the evaluation of $a \blacktriangledown b$ is composed of the evaluation of a followed by the evaluation of b in case a yields F and by the evaluation of b , but with a constant yield of T , in case a yields T . We can compute $\mathsf{FE}(a \blacktriangledown b)$ as follows.

$$\begin{aligned} \mathsf{FE}(a \blacktriangledown b) &= (\mathsf{T} \trianglelefteq a \trianglerighteq \mathsf{F})[\mathsf{T} \mapsto (\mathsf{T} \trianglelefteq b \trianglerighteq \mathsf{F})[\mathsf{F} \mapsto \mathsf{T}], \mathsf{F} \mapsto (\mathsf{T} \trianglelefteq b \trianglerighteq \mathsf{F})] \\ &= (\mathsf{T} \trianglelefteq a \trianglerighteq \mathsf{F})[\mathsf{T} \mapsto (\mathsf{T} \trianglelefteq b \trianglerighteq \mathsf{T}), \mathsf{F} \mapsto (\mathsf{T} \trianglelefteq b \trianglerighteq \mathsf{F})] \\ &= (\mathsf{T} \trianglelefteq b \trianglerighteq \mathsf{T}) \trianglelefteq a \trianglerighteq (\mathsf{T} \trianglelefteq b \trianglerighteq \mathsf{F}) \end{aligned}$$

Now the evaluation of $(a \vee b) \wedge c$ is a composition of this tree and $T \leq c \leq F$, as can be seen in Figure 2.1b.

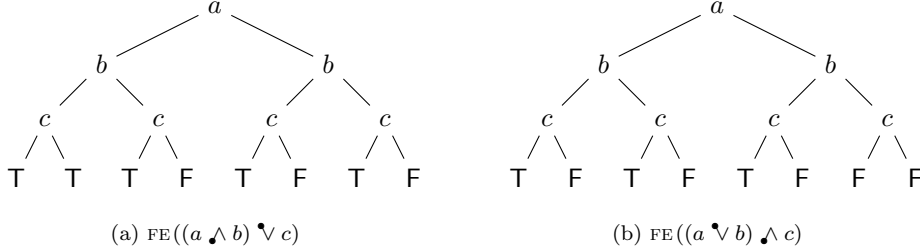


Figure 2.1: Trees depicting the evaluation of two FEL-terms. The evaluation starts at the root. When (the atom at) an inner node yields T the evaluation continues in its left branch and when it yields F it continues in its right branch. The leaves indicate the yield of the terms as a whole.

Informally we see that two FEL-terms are equal when they not only yield the same truth value given the truth values of their constituent atoms, but also contain the same atoms *in the same order*. Consider for example the terms a and $a \wedge (b \vee T)$ and note that the truth value of both is determined entirely by the truth value of a . Also note that since b occurs after a in the second term, no side effect of b could ever affect a . When both terms would be placed in the context of another term, e.g., $a \wedge c$ and $(a \wedge (b \vee T)) \wedge c$, the situation changes. A side effect of b might, for example, be that c will yield F . In that case the truth value of the first term is determined by a and c , while that of the second term is always F . We are now ready to define Fully Evaluated Left-Sequential Logic.

Definition 2.3. A **Fully Evaluated Left-Sequential Logic (FEL)** is a logic that satisfies the consequences of FE-equality. **Free Fully Evaluated Left-Sequential Logic (FFEL)** is the fully evaluated left-sequential logic that satisfies no more consequences than those of FE-equality, i.e., for all $P, Q \in FT$,

$$FEL \models P = Q \iff FE(P) = FE(Q) \quad \text{and} \quad FFEL \models P = Q \iff FE(P) = FE(Q).$$

It is not considered standard to define a logic equationally, but in this case we feel it is warranted to avoid having to mix the connectives from PL with those from FEL.

There is an immediate correspondence between trees and sets of traces, namely the paths of such trees annotated with the truth value of each internal node. For example, Figure 2.1a would correspond to the set of traces

$$\{(aTbTcT, T), (aTbTcF, T), (aTbFcT, T), (aTbFcF, F), \\ (aFbTcT, T), (aFbTcF, F), (aFbFcT, T), (aFbFcF, F)\}.$$

This means we could have defined the image of FE to be sets of such annotated traces. We chose to define FEL with tree semantics rather than with trace semantics because the former affords us a more succinct notation.

We now turn to the set of equations $EqFFEL$, listed in Table 2.1, which we will show in Section 2.3 is an axiomatization of FFEL. This set of equations was first presented by Blok in [Blo11].

$F = \neg T$	(FEL1)
$x \dot{\vee} y = \neg(\neg x \dot{\wedge} \neg y)$	(FEL2)
$\neg\neg x = x$	(FEL3)
$(x \dot{\wedge} y) \dot{\wedge} z = x \dot{\wedge} (y \dot{\wedge} z)$	(FEL4)
$T \dot{\wedge} x = x$	(FEL5)
$x \dot{\wedge} T = x$	(FEL6)
$x \dot{\wedge} F = F \dot{\wedge} x$	(FEL7)
$x \dot{\wedge} F = \neg x \dot{\wedge} F$	(FEL8)
$(x \dot{\wedge} F) \dot{\vee} y = (x \dot{\vee} T) \dot{\wedge} y$	(FEL9)
$x \dot{\vee} (y \dot{\wedge} F) = x \dot{\wedge} (y \dot{\vee} T)$	(FEL10)

Table 2.1: The set of equations **EqFFEL**.

If two FEL-terms s and t , possibly containing variables, are derivable by equational logic and EqFFEL, we denote this by $\text{EqFFEL} \vdash s = t$ and say that s and t are derivably equal. By virtue of (FEL1) through (FEL3), \blacktriangleleft is the dual of \blacktriangleright and hence the duals of the equations in EqFFEL are also derivable. We will use this fact implicitly throughout our proofs.

The following lemma shows some useful equations illustrating the special properties of terms of the form $x \blacktriangleleft F$ and $x \blacktriangleright T$. The first is an ‘extension’ of (FEL8) and the others show two different ways how terms of the form $x \blacktriangleright T$, and by duality terms of the form $x \blacktriangleleft F$, can change the main connective of a term.

Lemma 2.4. *The following equations can all be derived by equational logic and EqFFEL.*

1. $x \blacktriangleleft (y \blacktriangleleft F) = \neg x \blacktriangleleft (y \blacktriangleleft F)$
2. $(x \blacktriangleright T) \blacktriangleleft y = \neg(x \blacktriangleright T) \blacktriangleright y$
3. $x \blacktriangleright (y \blacktriangleleft (z \blacktriangleright T)) = (x \blacktriangleright y) \blacktriangleleft (z \blacktriangleright T)$

Proof. We derive the equations in order.

$$\begin{aligned}
x \blacktriangleleft (y \blacktriangleleft F) &= (x \blacktriangleleft F) \blacktriangleleft y && \text{by (FEL7) and (FEL4)} \\
&= (\neg x \blacktriangleleft F) \blacktriangleleft y && \text{by (FEL8)} \\
&= \neg x \blacktriangleleft (y \blacktriangleleft F) && \text{by (FEL7) and (FEL4)} \\
(x \blacktriangleright T) \blacktriangleleft y &= (x \blacktriangleleft F) \blacktriangleright y && \text{by (FEL9)} \\
&= (\neg x \blacktriangleleft F) \blacktriangleright y && \text{by (FEL8)} \\
&= (\neg x \blacktriangleleft \neg T) \blacktriangleright y && \text{by (FEL1)} \\
&= \neg(x \blacktriangleright T) \blacktriangleright y && \text{by (FEL3) and (FEL2)} \\
x \blacktriangleright (y \blacktriangleleft (z \blacktriangleright T)) &= x \blacktriangleright (y \blacktriangleright (z \blacktriangleleft F)) && \text{by (FEL10)} \\
&= (x \blacktriangleright y) \blacktriangleright (z \blacktriangleleft F) && \text{by the dual of (FEL4)} \\
&= (x \blacktriangleright y) \blacktriangleleft (z \blacktriangleright T) && \text{by (FEL10)} \quad \square
\end{aligned}$$

Theorem 2.5. *For all $P, Q \in \text{FT}$, if $\text{EqFFEL} \vdash P = Q$ then $\text{FFEL} \models P = Q$.*

Proof. It is immediately clear that identity, symmetry and transitivity are preserved. For congruence we show only that for all $P, Q, R \in \text{FT}$, $\text{FFEL} \models P = Q$ implies $\text{FFEL} \models R \blacktriangleleft P = R \blacktriangleleft Q$. The other cases proceed in a similar fashion. If $\text{FFEL} \models P = Q$, then $\text{FE}(P) = \text{FE}(Q)$ and hence $\text{FE}(P)[T \mapsto F] = \text{FE}(Q)[T \mapsto F]$, so

$$\text{FE}(R)[T \mapsto \text{FE}(P), F \mapsto \text{FE}(P)[T \mapsto F]] = \text{FE}(R)[T \mapsto \text{FE}(Q), F \mapsto \text{FE}(Q)[T \mapsto F]].$$

Therefore by definition of FE, $\text{FFEL} \models R \blacktriangleleft P = R \blacktriangleleft Q$.

The validity of the equations in EqFFEL is also easily verified. As an example we show this for (FEL8).

$$\begin{aligned}
\text{FE}(P \blacktriangleleft F) &= \text{FE}(P)[T \mapsto F, F \mapsto F[T \mapsto F]] && \text{by definition} \\
&= \text{FE}(P)[T \mapsto F] && \text{because } F[T \mapsto F] = F \\
&= \text{FE}(P)[T \mapsto F, F \mapsto T][T \mapsto F] && \text{by induction} \\
&= \text{FE}(\neg P \blacktriangleleft F),
\end{aligned}$$

where the induction that proves the third equality is on the structure of evaluation trees. \square

2.1 FEL Normal Form

To aid in our completeness proof we define a normal form for FEL-terms. Due to the possible presence of side effects, FFEL does not identify terms which contain different atoms or the same atoms in a different order. Because of this, common normal forms for PL are not normal forms for FEL-terms. For example, rewriting a term to Conjunctive Normal Form or Disjunctive Normal Form may require duplicating some of the atoms in the term, thus yielding a term that is not derivably equal to the original. We first present the grammar for our normal form, before motivating it. The normal form we present here is an adaptation of a normal form proposed by Blok in [Blo11].

Definition 2.6. *A term $P \in \text{FT}$ is said to be in **FEL Normal Form (FNF)** if it is generated by the following grammar.*

$$\begin{aligned}
P \in \text{FNF} &::= P^\top \mid P^F \mid P^\top \bullet \wedge P^* \\
P^* &::= P^c \mid P^d \\
P^c &::= P^\ell \mid P^* \bullet \wedge P^d \\
P^d &::= P^\ell \mid P^* \blacktriangledown P^c \\
P^\ell &::= a \bullet \wedge P^\top \mid \neg a \bullet \wedge P^\top \\
P^\top &::= \top \mid a \blacktriangledown P^\top \\
P^F &::= \text{F} \mid a \bullet \wedge P^F,
\end{aligned}$$

where $a \in A$. We refer to P^* -forms as $*$ -terms, to P^ℓ -forms as ℓ -terms, to P^\top -forms as \top -terms and to P^F -forms as F -terms. A term of the form $P^\top \bullet \wedge P^*$ is referred to as a \top -*-term.

We immediately note that if it were not for the presence of \top and F we could define a much simpler normal form. In that case it would suffice to ‘push in’ or ‘push down’ the negations, thus obtaining a Negation Normal Form, as exists for PL. Naturally if our set A of atoms is empty, the truth value constants would be a normal form.

When considering the image of FE we note that some trees only have \top -leaves, some only have F -leaves and some have both \top -leaves and F -leaves. For any FEL-term P , $\text{FE}(P \blacktriangledown \top)$ is a tree with only \top -leaves, as can easily be seen from the definition of FE. All terms P such that $\text{FE}(P)$ only has \top -leaves are rewritten to \top -terms. Similarly $\text{FE}(P \bullet \wedge F)$ is a tree with only F -leaves. All terms P such that $\text{FE}(P)$ only has F -leaves are rewritten to F -terms. The simplest trees in the image of FE that have both \top -leaves and F -leaves are $\text{FE}(a)$ for $a \in A$. Any (occurrence of an) atom that determines (in whole or in part) the yield of a term, such as a in this example, is referred to as a determinative (occurrence of an) atom. This as opposed to a non-determinative (occurrence of an) atom, such as the a in $a \bullet \wedge \top$, which does not determine (either in whole or in part) the yield of the term. Note that a term P such that $\text{FE}(P)$ contains both \top and F must contain at least one determinative atom.

Terms that contain at least one determinative atom will be rewritten to \top -*-terms. In \top -*-terms we encode each determinative atom together with the non-determinative atoms that occur between it and the next determinative atom in the term (reading from left to right) as an ℓ -term. Observe that the first atom in an ℓ -term is the (only) determinative atom in that ℓ -term and that determinative atoms only occur in ℓ -terms. Also observe that the yield of an ℓ -term is the yield of its determinative atom. This is intuitively convincing, because the remainder of the atoms in any ℓ -term are non-determinative and hence do not contribute to its yield. The non-determinative atoms that may occur before the first determinative atom are encoded as a

T-term. A T-*term is the conjunction of a T-term encoding such atoms and a *-term, which contains only conjunctions and disjunctions of ℓ -terms. We could also have encoded such atoms as an F-term and then taken the disjunction with a *-term to obtain a term with the same semantics. We consider ℓ -terms to be ‘basic’ in *-terms in the sense that they are the smallest grammatical unit that influences the yield of the *-term.

In the following, P^\top, P^ℓ , etc. are used both to denote grammatical categories and as variables for terms in those categories. The remainder of this section is concerned with defining and proving correct the normalization function $f : \text{FT} \rightarrow \text{FNF}$. We will define f recursively using the functions

$$f^n : \text{FNF} \rightarrow \text{FNF} \quad \text{and} \quad f^c : \text{FNF} \times \text{FNF} \rightarrow \text{FNF}.$$

The first of these will be used to rewrite negated FNF-terms to FNF-terms and the second to rewrite the conjunction of two FNF-terms to an FNF-term. By (FEL2) we have no need for a dedicated function that rewrites the disjunction of two FNF-terms to an FNF-term.

We start by defining f^n . Analyzing the semantics of T-terms and F-terms together with the definition of FE on negations, it becomes clear that f^n must turn T-terms into F-terms and vice versa. We also remark that f^n must preserve the left-associativity of the *-terms in T-*terms, modulo the associativity within ℓ -terms. We define $f^n : \text{FNF} \rightarrow \text{FNF}$ as follows, using the auxiliary function $f_1^n : P^* \rightarrow P^*$ to ‘push down’ or ‘push in’ the negation symbols when negating a T-*term. We note that there is no ambiguity between the different grammatical categories present in an FNF-term, i.e., any FNF-term is in exactly one of the grammatical categories identified in Definition 2.6.

$$f^n(\top) = \text{F} \tag{2.1}$$

$$f^n(a \blacktriangleright P^\top) = a \blacktriangleleft f^n(P^\top) \tag{2.2}$$

$$f^n(\text{F}) = \top \tag{2.3}$$

$$f^n(a \blacktriangleleft P^\text{F}) = a \blacktriangleright f^n(P^\text{F}). \tag{2.4}$$

$$f^n(P^\top \blacktriangleleft Q^*) = P^\top \blacktriangleleft f_1^n(Q^*) \tag{2.5}$$

$$f_1^n(a \blacktriangleleft P^\top) = \neg a \blacktriangleleft P^\top \tag{2.6}$$

$$f_1^n(\neg a \blacktriangleleft P^\top) = a \blacktriangleleft P^\top \tag{2.7}$$

$$f_1^n(P^* \blacktriangleleft Q^d) = f_1^n(P^*) \blacktriangleright f_1^n(Q^d) \tag{2.8}$$

$$f_1^n(P^* \blacktriangleright Q^c) = f_1^n(P^*) \blacktriangleleft f_1^n(Q^c) \tag{2.9}$$

Now we turn to defining f^c . These definitions have a great deal of inter-dependence so we first present the definition for f^c when the first argument is a T-term. We see that the conjunction of a T-term with another term always yields a term of the same grammatical category as the second conjunct.

$$f^c(\top, P) = P \tag{2.10}$$

$$f^c(a \blacktriangleright P^\top, Q^\top) = a \blacktriangleright f^c(P^\top, Q^\top) \tag{2.11}$$

$$f^c(a \blacktriangleright P^\top, Q^\text{F}) = a \blacktriangleleft f^c(P^\top, Q^\text{F}) \tag{2.12}$$

$$f^c(a \blacktriangleright P^\top, Q^\top \blacktriangleleft R^*) = f^c(a \blacktriangleright P^\top, Q^\top) \blacktriangleleft R^* \tag{2.13}$$

For defining f^c where the first argument is an F-term we make use of (FEL7) when dealing with conjunctions of F-terms with T-*terms. The definition of f^c for the arguments used in the right hand side of (2.16) starts at (2.23). We note that despite the high level of inter-dependence

in these definitions, this does not create a circular definition. We also note that the conjunction of an F-term with another term is always itself an F-term.

$$f^c(F, P^\top) = f^n(P^\top) \quad (2.14)$$

$$f^c(F, P^F) = P^F \quad (2.15)$$

$$f^c(F, P^\top \bullet Q^*) = f^c(P^\top \bullet Q^*, F) \quad (2.16)$$

$$f^c(a \bullet P^F, Q) = a \bullet f^c(P^F, Q) \quad (2.17)$$

The case where the first conjunct is a T-*term is the most complicated. Therefore we first consider the case where the second conjunct is a T-term. In this case we must make the T-term part of the last (rightmost) ℓ -term in the T-*term, so that the result will again be a T-*term. For this ‘pushing in’ of the second conjunct we define an auxiliary function $f_1^c : P^* \times P^\top \rightarrow P^*$.

$$f^c(P^\top \bullet Q^*, R^\top) = P^\top \bullet f_1^c(Q^*, R^\top) \quad (2.18)$$

$$f_1^c(a \bullet P^\top, Q^\top) = a \bullet f^c(P^\top, Q^\top) \quad (2.19)$$

$$f_1^c(\neg a \bullet P^\top, Q^\top) = \neg a \bullet f^c(P^\top, Q^\top) \quad (2.20)$$

$$f_1^c(P^* \bullet Q^d, R^\top) = P^* \bullet f_1^c(Q^d, R^\top) \quad (2.21)$$

$$f_1^c(P^* \bullet Q^c, R^\top) = P^* \bullet f_1^c(Q^c, R^\top) \quad (2.22)$$

When the second conjunct is an F-term, the result will naturally be an F-term itself. So we need to convert the T-*term to an F-term. Using (FEL4) we reduce this problem to converting a *-term to an F-term, for which we use the auxiliary function $f_2^c : P^* \times P^F \rightarrow P^F$.

$$f^c(P^\top \bullet Q^*, R^F) = f^c(P^\top, f_2^c(Q^*, R^F)) \quad (2.23)$$

$$f_2^c(a \bullet P^\top, R^F) = a \bullet f^c(P^\top, R^F) \quad (2.24)$$

$$f_2^c(\neg a \bullet P^\top, R^F) = a \bullet f^c(P^\top, R^F) \quad (2.25)$$

$$f_2^c(P^* \bullet Q^d, R^F) = f_2^c(P^*, f_2^c(Q^d, R^F)) \quad (2.26)$$

$$f_2^c(P^* \bullet Q^c, R^F) = f_2^c(P^*, f_2^c(Q^c, R^F)) \quad (2.27)$$

Finally we are left with conjunctions and disjunctions of two T-*terms, thus completing the definition of f^c . We use the auxiliary function $f_3^c : P^* \times P^\top \bullet P^* \rightarrow P^*$ to ensure that the result is a T-*term.

$$f^c(P^\top \bullet Q^*, R^\top \bullet S^*) = P^\top \bullet f_3^c(Q^*, R^\top \bullet S^*) \quad (2.28)$$

$$f_3^c(P^*, Q^\top \bullet R^\ell) = f_1^c(P^*, Q^\top) \bullet R^\ell \quad (2.29)$$

$$f_3^c(P^*, Q^\top \bullet (R^* \bullet S^d)) = f_3^c(P^*, Q^\top \bullet R^*) \bullet S^d \quad (2.30)$$

$$f_3^c(P^*, Q^\top \bullet (R^* \bullet S^c)) = f_1^c(P^*, Q^\top) \bullet (R^* \bullet S^c) \quad (2.31)$$

As promised, we now define the normalization function $f : \text{FT} \rightarrow \text{FNF}$ recursively, using f^n and f^c , as follows.

$$f(a) = \top \bullet (a \bullet \top) \quad (2.32)$$

$$f(\top) = \top \quad (2.33)$$

$$f(F) = F \quad (2.34)$$

$$f(\neg P) = f^n(f(P)) \quad (2.35)$$

$$f(P \bullet Q) = f^c(f(P), f(Q)) \quad (2.36)$$

$$f(P \bullet Q) = f^n(f^c(f^n(f(P)), f^n(f(Q)))) \quad (2.37)$$

Theorem 2.7. *For any $P \in \text{FT}$, $f(P)$ terminates, $f(P) \in \text{FNF}$ and $\text{EqFFEL} \vdash f(P) = P$.*

In Appendix A.1 we first prove a number of lemmas showing that the definitions f^n and f^c are correct and use those to prove the theorem. The reader might wonder why we have used a normalization function rather than a term rewriting system to prove the correctness of FNF. The main reason is the author's lack of experience with term rewriting systems, although the fact that using a function relieves us of the need to prove the confluence of the induced rewriting system, thus simplifying the proof, is also a factor.

2.2 Tree Structure

In Section 2.3 we will prove that EqFFEL axiomatizes FFEL by showing that for $P \in \text{FNF}$ we can invert $\text{FE}(P)$. To do this we need to prove several structural properties of the trees in the image of FE. In the definition of FE we can see how $\text{FE}(P \blacktriangle Q)$ is assembled from $\text{FE}(P)$ and $\text{FE}(Q)$ and similarly for $\text{FE}(P \blacktriangledown Q)$. To decompose these trees we will introduce some notation. The trees in the image of FE are all finite binary trees over A with leaves in $\{\text{T}, \text{F}\}$, i.e., $\text{FE}[\text{FT}] \subseteq \mathcal{T}$. We will now also consider the set \mathcal{T}_\square of binary trees over A with leaves in $\{\text{T}, \text{F}, \square\}$, where the ' \square ' symbol is pronounced 'box'. Similarly we consider $\mathcal{T}_{1,2}$, the set of binary trees over A with leaves in $\{\text{T}, \text{F}, \square_1, \square_2\}$. The \square , \square_1 and \square_2 will be used as placeholders when composing or decomposing trees. Replacement of the leaves of trees in \mathcal{T}_\square and $\mathcal{T}_{1,2}$ by trees (either in \mathcal{T} , \mathcal{T}_\square or $\mathcal{T}_{1,2}$) is defined analogous to replacement for trees in \mathcal{T} , adopting the same notational conventions.

For example we have by definition of FE that $\text{FE}(P \blacktriangle Q)$ can be decomposed as

$$\text{FE}(P)[\text{T} \mapsto \square_1, \text{F} \mapsto \square_2][\square_1 \mapsto \text{FE}(Q), \square_2 \mapsto \text{FE}(Q)[\text{T} \mapsto \text{F}]],$$

where $\text{FE}(P)[\text{T} \mapsto \square_1, \text{F} \mapsto \square_2] \in \mathcal{T}_{1,2}$ and $\text{FE}(Q)$ and $\text{FE}(Q)[\text{T} \mapsto \text{F}]$ are in \mathcal{T} . We note that this only works because the trees in the image of FE, or more general, in \mathcal{T} , do not contain any boxes. Similarly, as we discussed previously, $\text{FE}(P \blacktriangle \text{F}) = \text{FE}(P)[\text{T} \mapsto \text{F}]$, which we can write as $\text{FE}(P)[\text{T} \mapsto \square][\square \mapsto \text{F}]$. We start by analyzing the FE-image of ℓ -terms.

Lemma 2.8 (Structure of ℓ -terms). *There is no ℓ -term P such that $\text{FE}(P)$ can be decomposed as $X[\square \mapsto Y]$ with $X \in \mathcal{T}_\square$ and $Y \in \mathcal{T}$, where $X \neq \square$, but does contain \square , and Y contains occurrences of both T and F .*

Proof. Let P be some ℓ -term. When we analyze the grammar of P we find that one branch from the root of $\text{FE}(P)$ will only contain T and not F and the other branch vice versa. Hence if $\text{FE}(P) = X[\square \mapsto Y]$ and Y contains occurrences of both T and F , then Y must contain the root and hence $X = \square$. \square

By definition a $*$ -term contains at least one ℓ -term and hence for any $*$ -term P , $\text{FE}(P)$ contains both T and F . The following lemma provides the FE-image of the rightmost ℓ -term in a $*$ -term to witness this fact.

Lemma 2.9 (Determinativeness). *For all $*$ -terms P , $\text{FE}(P)$ can be decomposed as $X[\square \mapsto Y]$ with $X \in \mathcal{T}_\square$ and $Y \in \mathcal{T}$ such that X contains \square and $Y = \text{FE}(Q)$ for some ℓ -term Q . Note that X may be \square . We will refer to Y as the witness for this lemma for P .*

Proof. By induction on the complexity of $*$ -terms P modulo the complexity of ℓ -terms. In the base case P is an ℓ -term and $\text{FE}(P) = \square[\square \mapsto \text{FE}(P)]$ is the desired decomposition by Lemma 2.8. For the induction we have to consider both $\text{FE}(P \blacktriangle Q)$ and $\text{FE}(P \blacktriangledown Q)$.

We treat only the case for $\text{FE}(P \blacktriangleleft Q)$, the case for $\text{FE}(P \blacktriangleright Q)$ is analogous. Let $X[\Box \mapsto Y]$ be the decomposition for $\text{FE}(Q)$ which we have by induction hypothesis. Since by definition of FE on \blacktriangleleft we have

$$\text{FE}(P \blacktriangleleft Q) = \text{FE}(P)[\text{T} \mapsto \text{FE}(Q), \text{F} \mapsto \text{FE}(Q)[\text{T} \mapsto \text{F}]],$$

we also have

$$\begin{aligned} \text{FE}(P \blacktriangleleft Q) &= \text{FE}(P)[\text{T} \mapsto X[\Box \mapsto Y], \text{F} \mapsto \text{FE}(Q)[\text{T} \mapsto \text{F}]] \\ &= \text{FE}(P)[\text{T} \mapsto X, \text{F} \mapsto \text{FE}(Q)[\text{T} \mapsto \text{F}]][\Box \mapsto Y], \end{aligned}$$

where the second equality is due to the fact that the only boxes in

$$\text{FE}(P)[\text{T} \mapsto X, \text{F} \mapsto \text{FE}(Q)[\text{T} \mapsto \text{F}]]$$

are those occurring in X . This gives our desired decomposition. \square

The following lemma illustrates another structural property of trees in the image of $*$ -terms under FE , namely that the left branch of any determinative atom in such a tree is different from its right branch.

Lemma 2.10 (Non-decomposition). *There is no $*$ -term P such that $\text{FE}(P)$ can be decomposed as $X[\Box \mapsto Y]$ with $X \in \mathcal{T}_\Box$ and $Y \in \mathcal{T}$, where $X \neq \Box$ and X contains \Box , but not T or F .*

Proof. By induction on P modulo the complexity of ℓ -terms. The base case covers ℓ -terms and follows immediately from Lemma 2.9 ($\text{FE}(P)$ contains occurrences of both T and F) and Lemma 2.8 (no non-trivial decomposition exists that contains both). For the induction we assume that the lemma holds for all $*$ -terms with lesser complexity than $P \blacktriangleleft Q$ and $P \blacktriangleright Q$.

We start with the case for $\text{FE}(P \blacktriangleleft Q)$. Suppose for contradiction that $\text{FE}(P \blacktriangleleft Q) = X[\Box \mapsto Y]$ with $X \neq \Box$ and X not containing any occurrences of T or F . Let R be a witness of Lemma 2.9 for P . Now note that $\text{FE}(P \blacktriangleleft Q)$ has a subtree

$$R[\text{T} \mapsto \text{FE}(Q), \text{F} \mapsto \text{FE}(Q)[\text{T} \mapsto \text{F}]].$$

Because Y must contain both the occurrences of F in the one branch from the root of this subtree as well as the occurrences of $\text{FE}(Q)$ in the other (because they contain T and F), Lemma 2.8 implies that Y must (strictly) contain $\text{FE}(Q)$ and $\text{FE}(Q)[\text{T} \mapsto \text{F}]$. Hence there is a $Z \in \mathcal{T}$ such that $\text{FE}(P) = X[\Box \mapsto Z]$, which violates the induction hypothesis. The case for $\text{FE}(P \blacktriangleright Q)$ proceeds analogously. \square

We now arrive at two crucial definitions for our completeness proof. When considering $*$ -terms we already know that $\text{FE}(P \blacktriangleleft Q)$ can be decomposed as

$$\text{FE}(P)[\text{T} \mapsto \Box_1, \text{F} \mapsto \Box_2][\Box_1 \mapsto \text{FE}(Q), \Box_2 \mapsto \text{FE}(Q)[\text{T} \mapsto \text{F}]].$$

Our goal now is to give a definition for a type of decomposition so that this is the only such decomposition for $\text{FE}(P \blacktriangleleft Q)$. We also ensure that $\text{FE}(P \blacktriangleright Q)$ does not have a decomposition of that type, so that we can distinguish $\text{FE}(P \blacktriangleleft Q)$ from $\text{FE}(P \blacktriangleright Q)$. Similarly, we define another type of decomposition so that $\text{FE}(P \blacktriangleright Q)$ can only be decomposed as

$$\text{FE}(P)[\text{T} \mapsto \Box_1, \text{F} \mapsto \Box_2][\Box_1 \mapsto \text{FE}(Q)[\text{F} \mapsto \text{T}], \Box_2 \mapsto \text{FE}(Q)]$$

and that $\text{FE}(P \blacktriangleleft Q)$ does not have a decomposition of that type.

Definition 2.11. The pair $(Y, Z) \in \mathcal{T}_{1,2} \times \mathcal{T}$ is a **candidate conjunction decomposition (ccd)** of $X \in \mathcal{T}$, if

- $X = Y[\Box_1 \mapsto Z, \Box_2 \mapsto Z[\top \mapsto \text{F}]]$,
- Y contains both \Box_1 and \Box_2 ,
- Y contains neither \top nor F , and
- Z contains both \top and F .

Similarly, (Y, Z) is a **candidate disjunction decomposition (cdd)** of X , if

- $X = Y[\Box_1 \mapsto Z[\text{F} \mapsto \top], \Box_2 \mapsto Z]$,
- Y contains both \Box_1 and \Box_2 ,
- Y contains neither \top nor F , and
- Z contains both \top and F .

The ccd and cdd are not necessarily the decompositions we are looking for, because, for example, $\text{FE}((P \blacktriangleleft Q) \blacktriangleleft R)$ has a ccd $(\text{FE}(P)[\top \mapsto \Box_1, \text{F} \mapsto \Box_2], \text{FE}(Q \blacktriangleleft R))$, whereas the decomposition we need is $(\text{FE}(P \blacktriangleleft Q)[\top \mapsto \Box_1, \text{F} \mapsto \Box_2], \text{FE}(R))$. Therefore we refine these definitions to obtain the decompositions we seek.

Definition 2.12. The pair $(Y, Z) \in \mathcal{T}_{1,2} \times \mathcal{T}$ is a **conjunction decomposition (cd)** of $X \in \mathcal{T}$, if it is a ccd of X and there is no other ccd (Y', Z') of X where the depth of Z' is smaller than that of Z . Similarly, (Y, Z) is a **disjunction decomposition (dd)** of X , if it is a cdd of X and there is no other cdd (Y', Z') of X where the depth of Z' is smaller than that of Z .

Theorem 2.13. For any $*$ -term $P \blacktriangleleft Q$, i.e., with $P \in P^*$ and $Q \in P^d$, $\text{FE}(P \blacktriangleleft Q)$ has the (unique) cd

$$(\text{FE}(P)[\top \mapsto \Box_1, \text{F} \mapsto \Box_2], \text{FE}(Q))$$

and no dd. For any $*$ -term $P \blacktriangledown Q$, i.e., with $P \in P^*$ and $Q \in P^c$, $\text{FE}(P \blacktriangledown Q)$ has no cd and its (unique) dd is

$$(\text{FE}(P)[\top \mapsto \Box_1, \text{F} \mapsto \Box_2], \text{FE}(Q)).$$

Proof. We first treat the case for $P \blacktriangleleft Q$ and start with cd. Note that $\text{FE}(P \blacktriangleleft Q)$ has a ccd $(\text{FE}(P)[\top \mapsto \Box_1, \text{F} \mapsto \Box_2], \text{FE}(Q))$ by definition of FE (for the first condition) and by Lemma 2.9 (for the fourth condition). It is immediate that it satisfies the second and third conditions. It also follows that for any ccd (Y, Z) either Z contains or is contained in $\text{FE}(Q)$, for suppose otherwise, then Y will contain an occurrence of \top or of F , namely those we know by Lemma 2.9 that $\text{FE}(Q)$ has. Therefore it suffices to show that there is no ccd (Y, Z) where Z is strictly contained in $\text{FE}(Q)$. Suppose for contradiction that (Y, Z) is such a ccd. If Z is strictly contained in $\text{FE}(Q)$ we can decompose $\text{FE}(Q)$ as $\text{FE}(Q) = U[\Box \mapsto Z]$ for some $U \in \mathcal{T}_\Box$ that contains but is not equal to \Box . By Lemma 2.10 this implies that U contains \top or F . But then so does Y , because

$$Y = \text{FE}(P)[\top \mapsto U[\Box \mapsto \Box_1], \text{F} \mapsto U[\Box \mapsto \Box_2]],$$

and so (Y, Z) is not a ccd for $\text{FE}(P \blacktriangleleft Q)$. Therefore $(\text{FE}(P)[\top \mapsto \Box_1, \text{F} \mapsto \Box_2], \text{FE}(Q))$ is the *unique* cd for $\text{FE}(P \blacktriangleleft Q)$.

Now for the dd. It suffices to show that there is no cdd for $\text{FE}(P \blacktriangleleft Q)$. Suppose for contradiction that (Y, Z) is a cdd for $\text{FE}(P \blacktriangleleft Q)$. We note that Z cannot be contained in $\text{FE}(Q)$, for then by Lemma 2.10 Y would contain \top or F . So Z (strictly) contains $\text{FE}(Q)$. But then because

$$Y[\Box_1 \mapsto Z[\text{F} \mapsto \top], \Box_2 \mapsto Z] = \text{FE}(P \blacktriangleleft Q),$$

we would have by Lemma 2.9 that $\text{FE}(P \blacktriangleleft Q)$ does not contain an occurrence of $\text{FE}(Q)[\top \mapsto \text{F}]$. But the cd of $\text{FE}(P \blacktriangleleft Q)$ tells us that it does, contradiction! Therefore there is no cdd, and hence no dd, for $\text{FE}(P \blacktriangleleft Q)$. The case for $\text{FE}(P \blacktriangleright Q)$ proceeds analogously. \square

At this point we have the tools necessary to invert FE on $*$ -terms, at least down to the level of ℓ -terms. We note that we can easily detect if a tree in the image of FE is in the image of P^ℓ , because all leaves to the left of the root are one truth value, while all the leaves to the right are the other. To invert FE on \top - $*$ -terms we still need to be able to reconstruct $\text{FE}(P^\top)$ and $\text{FE}(Q^*)$ from $\text{FE}(P^\top \blacktriangleleft Q^*)$. To this end we define a \top - $*$ -decomposition.

Definition 2.14. *The pair $(Y, Z) \in \mathcal{T}_\square \times \mathcal{T}$ is a \top - $*$ -decomposition (tsd) of $X \in \mathcal{T}$, if $X = Y[\square \mapsto Z]$, Y does not contain \top or F and there is no decomposition $(U, V) \in \mathcal{T}_\square \times \mathcal{T}$ of Z such that*

- $Z = U[\square \mapsto V]$,
- U contains \square ,
- $U \neq \square$, and
- U contains neither \top nor F .

Theorem 2.15. *For any \top -term P and $*$ -term Q the (unique) tsd of $\text{FE}(P \blacktriangleleft Q)$ is*

$$(\text{FE}(P)[\top \mapsto \square], \text{FE}(Q)).$$

Proof. First we observe that $(\text{FE}(P)[\top \mapsto \square], \text{FE}(Q))$ is a tsd because by definition of FE on \blacktriangleleft we have $\text{FE}(P)[\top \mapsto \text{FE}(Q)] = \text{FE}(P \blacktriangleleft Q)$ and $\text{FE}(Q)$ is non-decomposable by Lemma 2.10.

Suppose for contradiction that there is another tsd (Y, Z) of $\text{FE}(P \blacktriangleleft Q)$. Now Z must contain or be contained in $\text{FE}(Q)$ for otherwise Y would contain \top or F , i.e., the ones we know $\text{FE}(Q)$ has by Lemma 2.9.

If Z is strictly contained in $\text{FE}(Q)$, then $\text{FE}(Q) = U[\square \mapsto Z]$ for some $U \in \mathcal{T}_\square$ with $U \neq \square$ and U not containing \top or F (because then Y would too). But this violates Lemma 2.10, which states that no such decomposition exists. If Z strictly contains $\text{FE}(Q)$, then Z contains at least one atom from P . But the left branch of any atom in $\text{FE}(P)$ is equal to its right branch and hence Z is decomposable. Therefore $(\text{FE}(P)[\top \mapsto \square], \text{FE}(Q))$ is the *unique* tsd of $\text{FE}(P \blacktriangleleft Q)$. \square

2.3 Completeness

With the two theorems from the previous section, we can prove completeness for FFEL. We define three auxiliary functions to aid in our definition of the inverse of FE on FNF. Let $\text{cd} : \mathcal{T} \rightarrow \mathcal{T}_{1,2} \times \mathcal{T}$ be the function that returns the conjunction decomposition of its argument, dd of the same type its disjunction decomposition and $\text{tsd} : \mathcal{T} \rightarrow \mathcal{T}_\square \times \mathcal{T}$ its \top - $*$ -decomposition. Naturally, these functions are undefined when their argument does not have a decomposition of the specified type. Each of these functions returns a pair and we will use cd_1 (dd_1 , tsd_1) to denote the first element of this pair and cd_2 (dd_2 , tsd_2) to denote the second element.

We define $g : \mathcal{T} \rightarrow \text{FT}$ using the functions $g^\top : \mathcal{T} \rightarrow \text{FT}$ for inverting trees in the image of \top -terms and g^F , g^ℓ and g^* of the same type for inverting trees in the image of F -terms, ℓ -terms and $*$ -terms, respectively. These functions are defined as follows.

$$g^\top(X) = \begin{cases} \top & \text{if } X = \top \\ a \blacktriangleright g^\top(Y) & \text{if } X = Y \trianglelefteq a \triangleright Z \end{cases} \quad (2.38)$$

We note that we might as well have used the right branch from the root in the recursive case. We chose the left branch here to more closely mirror the definition of the corresponding function for FSCL_{SE} , defined in Chapter 3.

$$g^F(X) = \begin{cases} F & \text{if } X = F \\ a \blacktriangleleft g^F(Z) & \text{if } X = Y \trianglelefteq a \trianglerighteq Z \end{cases} \quad (2.39)$$

Similarly, we could have taken the left branch in this case.

$$g^\ell(X) = \begin{cases} a \blacktriangleleft g^\ell(Y) & \text{if } X = Y \trianglelefteq a \trianglerighteq Z \text{ for some } a \in A \\ & \text{and } Y \text{ only has T-leaves} \\ \neg a \blacktriangleleft g^\ell(Z) & \text{if } X = Y \trianglelefteq a \trianglerighteq Z \text{ for some } a \in A \\ & \text{and } Z \text{ only has T-leaves} \end{cases} \quad (2.40)$$

$$g^*(X) = \begin{cases} g^*(\text{cd}_1(X)[\Box_1 \mapsto \text{T}, \Box_2 \mapsto \text{F}]) \blacktriangleleft g^*(\text{cd}_2(X)) & \text{if } X \text{ has a cd} \\ g^*(\text{dd}_1(X)[\Box_1 \mapsto \text{T}, \Box_2 \mapsto \text{F}]) \blacktriangleright g^*(\text{dd}_2(X)) & \text{if } X \text{ has a dd} \\ g^\ell(X) & \text{otherwise} \end{cases} \quad (2.41)$$

We can immediately see how Theorem 2.13 will be used in the correctness proof of g^* .

$$g(X) = \begin{cases} g^\ell(X) & \text{if } X \text{ has only T-leaves} \\ g^F(X) & \text{if } X \text{ has only F-leaves} \\ g^\ell(\text{tsd}_1(X)[\Box \mapsto \text{T}]) \blacktriangleleft g^*(\text{tsd}_2(X)) & \text{otherwise} \end{cases} \quad (2.42)$$

Similarly, we can see how Theorem 2.15 is used in the correctness proof of g . It should come as no surprise that g is indeed correct and inverts FE on FNF.

Theorem 2.16. *For all $P \in \text{FNF}$, $g(\text{FE}(P)) \equiv P$.*

The proof for this theorem can be found in Appendix A.2. For the sake of completeness, we separately state the completeness result below.

Theorem 2.17. *For all $P, Q \in \text{FT}$, if $\text{FFEL} \models P = Q$ then $\text{EqFFEL} \vdash P = Q$.*

Proof. It suffices to show that for $P, Q \in \text{FNF}$, $\text{FE}(P) = \text{FE}(Q)$ implies $P \equiv Q$. To see this suppose that P' and Q' are two FEL-terms and $\text{FE}(P') = \text{FE}(Q')$. We know that P' is derivably equal to an FNF-term P , i.e., $\text{EqFFEL} \vdash P' = P$, and that Q' is derivably equal to an FNF-term Q , i.e., $\text{EqFFEL} \vdash Q' = Q$. Theorem 2.5 then gives us $\text{FE}(P') = \text{FE}(P)$ and $\text{FE}(Q') = \text{FE}(Q)$. Hence by the result $P \equiv Q$ and in particular $\text{EqFFEL} \vdash P = Q$. Transitivity then gives us $\text{EqFFEL} \vdash P' = Q'$ as desired.

The result follows immediately from Theorem 2.16. \square

Free Short-Circuit Logic (FSCL)

In this chapter we define Free Short-Circuit Logic on evaluation trees and present the set of equations EqFSCL, which we will prove axiomatizes this logic in Section 2.3. Formally, SCL-terms are built up from atomic propositions that may have side effects, called atoms, the truth value constants **T** for true and **F** for false and the connectives \neg for negation, $\mathbin{\text{\textcircled{\tiny \wedge}}}$ for (short-circuit) left-sequential conjunction and $\mathbin{\text{\textcircled{\tiny \vee}}}$ for (short-circuit) left-sequential disjunction.

Definition 3.1. *Let A be a countable set of atoms. SCL-terms (ST) have the following grammar presented in Backus-Naur Form.*

$$P \in \text{ST} ::= a \in A \mid \mathbf{T} \mid \mathbf{F} \mid \neg P \mid (P \mathbin{\text{\textcircled{\tiny \wedge}}} P) \mid (P \mathbin{\text{\textcircled{\tiny \vee}}} P)$$

As is the case with FEL, if $A = \emptyset$ then resulting logic is trivial.

First we return for a moment to our motivation for left-sequential logics, i.e., propositional terms as used in programming languages. We will consider the SCL-term $a \mathbin{\text{\textcircled{\tiny \vee}}} b$ and informally describe its evaluation, naturally using a short-circuit evaluation strategy. We start by evaluating a and let its yield determine our next action. If a yielded **F** we proceed by evaluating b , i.e., the yield of the term as a whole will be the yield of b . If a yielded **T**, we already know at this point that $a \mathbin{\text{\textcircled{\tiny \vee}}} b$ will yield **T**. We skip the evaluation of b and let the term yield **T**, i.e., b is short-circuited.

Considering the more complex term $(a \mathbin{\text{\textcircled{\tiny \vee}}} b) \mathbin{\text{\textcircled{\tiny \wedge}}} c$, we find that we start by evaluating $a \mathbin{\text{\textcircled{\tiny \vee}}} b$ and if it yields **T** we proceed by evaluating c . If it yields **F** we skip the evaluation of c , because we know the term will yield **F**. This example shows that evaluating SCL-terms is an interactive procedure, where the yield of the previous atom is needed to determine which atom to evaluate next. We believe these semantics are best captured in trees. Hence we will define equality of SCL-terms using (evaluation) trees. We define the set \mathcal{T} of finite binary trees over A with leaves in $\{\mathbf{T}, \mathbf{F}\}$ recursively. We have that

$$\mathbf{T} \in \mathcal{T}, \quad \mathbf{F} \in \mathcal{T}, \quad \text{and} \quad (X \trianglelefteq a \trianglerighteq Y) \in \mathcal{T} \text{ for any } X, Y \in \mathcal{T} \text{ and } a \in A.$$

In the expression $X \trianglelefteq a \trianglerighteq Y$ the root is represented by a , the left branch by X and the right branch by Y . We define the depth of a tree X recursively by $d(\mathbf{T}) = d(\mathbf{F}) = 0$ and $d(X \trianglelefteq a \trianglerighteq Z) = 1 + \max(d(X), d(Z))$ for $a \in A$. The reason for our choice of notation for trees will become apparent in Chapter 4. We refer to trees in \mathcal{T} as evaluation trees, or trees for short. Figure 3.1 shows the trees corresponding to the evaluations of $(a \mathbin{\text{\textcircled{\tiny \vee}}} b) \mathbin{\text{\textcircled{\tiny \wedge}}} c$ and $(a \mathbin{\text{\textcircled{\tiny \wedge}}} b) \mathbin{\text{\textcircled{\tiny \vee}}} c$.

Returning to our example, we have seen that the tree corresponding to the evaluation of $(a \circlearrowleft b) \triangleleft c$ can be composed from the tree corresponding to the evaluation of $a \circlearrowleft b$ and that corresponding to the evaluation of c . We said above that if $a \circlearrowleft b$ yielded \top , we would proceed with the evaluation of c . This can be seen as replacing each \top -leaf in the tree corresponding to the evaluation of $a \circlearrowleft b$ with the tree that corresponds to the evaluation of c . Formally we define the leaf replacement operator, ‘replacement’ for short, on trees in \mathcal{T} as follows. Let $X, X', X'', Y, Z \in \mathcal{T}$ and $a \in A$. The replacement of \top with Y and \bot with Z in X , denoted $X[\top \mapsto Y, \bot \mapsto Z]$, is defined recursively as

$$\begin{aligned} \top[\top \mapsto Y, \bot \mapsto Z] &= Y \\ \bot[\top \mapsto Y, \bot \mapsto Z] &= Z \\ (X' \triangleleft a \triangleright X'')[\top \mapsto Y, \bot \mapsto Z] &= X'[\top \mapsto Y, \bot \mapsto Z] \triangleleft a \triangleright X''[\top \mapsto Y, \bot \mapsto Z]. \end{aligned}$$

We note that the order in which the replacements of the leaves of X is listed inside the brackets is irrelevant. We will adopt the convention of not listing any identities inside the brackets, i.e.,

$$X[\bot \mapsto Y] = X[\top \mapsto \top, \bot \mapsto Y].$$

Furthermore we let replacements associate to the left. We also use that fact that

$$X[\top \mapsto Y][\bot \mapsto Z] = X[\top \mapsto Y, \bot \mapsto Z]$$

if Y does not contain \bot , which can be shown by a trivial induction. Similarly,

$$X[\bot \mapsto Z][\top \mapsto Y] = X[\top \mapsto Y, \bot \mapsto Z]$$

if Z does not contain \top . We now have the terminology and notation to formally define the mapping from SCL-terms to evaluation trees.

Definition 3.2. *Let A be a countable set of atoms and let \mathcal{T} be the set of all finite binary trees over A with leaves in $\{\top, \bot\}$. We define the unary **Short-Circuit Evaluation** function $\text{SE} : \text{ST} \rightarrow \mathcal{T}$ as:*

$$\begin{aligned} \text{SE}(\top) &= \top \\ \text{SE}(\bot) &= \bot \\ \text{SE}(a) &= \top \triangleleft a \triangleright \bot && \text{for } a \in A \\ \text{SE}(\neg P) &= \text{SE}(P)[\top \mapsto \bot, \bot \mapsto \top] \\ \text{SE}(P \triangleleft Q) &= \text{SE}(P)[\top \mapsto \text{SE}(Q)] \\ \text{SE}(P \circlearrowleft Q) &= \text{SE}(P)[\bot \mapsto \text{SE}(Q)]. \end{aligned}$$

As we can see from the definition on atoms, the evaluation continues in the left branch if an atom yields \top and in the right branch if it yields \bot . Revisiting our example once more, we indeed see how the evaluation of $a \circlearrowleft b$ is composed of the evaluation of a followed by the evaluation of b in case a yields \bot . We can compute

$$\begin{aligned} \text{SE}(a \circlearrowleft b) &= (\top \triangleleft a \triangleright \bot)[\bot \mapsto (\top \triangleleft b \triangleright \bot)] \\ &= \top \triangleleft a \triangleright (\top \triangleleft b \triangleright \bot). \end{aligned}$$

Now the evaluation of $(a \circlearrowleft b) \triangleleft c$ is a composition of this tree and $\top \triangleleft c \triangleright \bot$, as can be seen in Figure 2.1b.

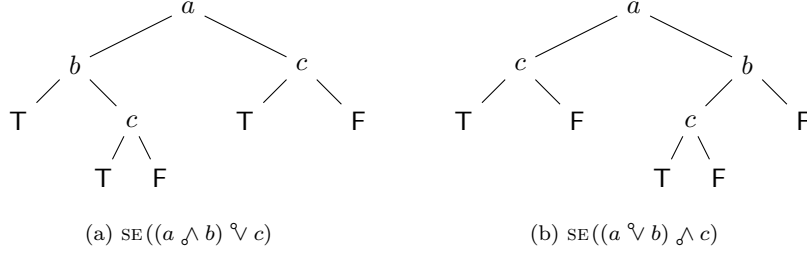


Figure 3.1: Trees depicting the evaluation of two SCL-terms. The evaluation starts at the root. When (the atom at) an inner node yields T the evaluation continues in its left branch and when it yields F it continues in its right branch. The leaves indicate the yield of the terms as a whole.

These trees show us that a function of the yield of the atoms in an SCL-term is insufficient to determine the semantics of the term as a whole. They show us that we must also consider the (conditional) order in which the atoms occur in the term. In particular we see that in $P \vee Q$, Q will be short-circuited if P yields T, while in $P \wedge Q$, it will be short-circuited if P yields F. We are now ready to define Short-Circuit Logic on evaluation trees.

Definition 3.3. Free Short-Circuit Logic (FSCL_{SE}) is the logic that satisfies exactly the consequences of SE-equality, i.e., for all $P, Q \in \text{ST}$,

$$\text{FSCL}_{\text{SE}} \models P = Q \iff \text{SE}(P) = \text{SE}(Q).$$

Using the completeness result we shall prove in Section 3.3, we will show that FSCL_{SE} is in fact equivalent to FSCL as defined by Bergstra and Ponse in [BP10b]. This should come as no surprise given the tree-like structure that Proposition Algebra terms exhibit, see, e.g., [BP11] or [BP12].

We choose a representation of \mathcal{T} as trees rather than as sets of traces, i.e., the paths of those trees annotated with truth values for the atoms, because the tree notation allows us to be more succinct. These tree semantics were first given, although presented as trace semantics, by Ponse in [Pon11].

We now turn to the set of equations EqFSCL , listed in Table 3.1, which we will show in Section 3.3 is an axiomatization of FSCL_{SE} . This set of equations is based on one presented by Bergstra and Ponse in [BP10b]. If two SCL-terms s and t , where we extend the definition to allow for terms containing variables, are derivable by equational logic and EqFSCL , we denote this by $\text{EqFSCL} \vdash s = t$ and say that s and t are derivably equal. As a consequence of (SCL1) through (SCL3), \wedge is the dual of \vee and hence the duals of the equations in EqFSCL are also derivable. We will use this fact implicitly throughout our proofs. Observe that unlike with EqFFEL , we have an equation in EqFSCL for (a special case of) distributivity, i.e., (SCL10).

The following lemma shows some equations that will prove useful in Section 3.1. These equations show how terms of the form $x \wedge \text{F}$ and $x \vee \text{T}$ can be used to change the order in which atoms occur in an SCL-term. This is very different from the situation with FEL, where terms that contain the same atoms, but in a different order, are never derivably equal. In terms of a comparison between EqFSCL and EqFFEL this can be seen as a consequence of (SCL10).

$\mathbf{F} = \neg \mathbf{T}$	(SCL1)
$x \circlearrowleft y = \neg(\neg x \circlearrowright \neg y)$	(SCL2)
$\neg\neg x = x$	(SCL3)
$(x \circlearrowright y) \circlearrowright z = x \circlearrowright (y \circlearrowright z)$	(SCL4)
$\mathbf{T} \circlearrowright x = x$	(SCL5)
$x \circlearrowright \mathbf{T} = x$	(SCL6)
$\mathbf{F} \circlearrowright x = \mathbf{F}$	(SCL7)
$x \circlearrowright \mathbf{F} = \neg x \circlearrowright \mathbf{F}$	(SCL8)
$(x \circlearrowright \mathbf{F}) \circlearrowleft y = (x \circlearrowleft \mathbf{T}) \circlearrowright y$	(SCL9)
$(x \circlearrowright y) \circlearrowleft (z \circlearrowright \mathbf{F}) = (x \circlearrowleft (z \circlearrowright \mathbf{F})) \circlearrowright (y \circlearrowleft (z \circlearrowright \mathbf{F}))$	(SCL10)

Table 3.1: The set of equations **EqFSCL**.

Lemma 3.4. *The following equations can all be derived by equational logic and EqFSCL.*

1. $(x \circlearrowleft y) \circlearrowright (z \circlearrowright F) = (\neg x \circlearrowleft (z \circlearrowright F)) \circlearrowright (y \circlearrowright (z \circlearrowright F))$
2. $(x \circlearrowleft (y \circlearrowright F)) \circlearrowright (z \circlearrowright F) = (\neg x \circlearrowleft (z \circlearrowright F)) \circlearrowright (y \circlearrowright F)$
3. $(x \circlearrowright (y \circlearrowleft T)) \circlearrowleft (z \circlearrowright F) = (x \circlearrowleft (z \circlearrowright F)) \circlearrowright (y \circlearrowleft T)$

Proof. We derive the equations in order.

$$\begin{aligned}
& (x \circlearrowleft y) \circlearrowright (z \circlearrowright F) \\
&= (x \circlearrowleft y) \circlearrowright ((z \circlearrowright F) \circlearrowright F) && \text{by (SCL7) and (SCL4)} \\
&= (x \circlearrowleft y) \circlearrowright (\neg(z \circlearrowright F) \circlearrowright F) && \text{by (SCL8)} \\
&= ((x \circlearrowleft y) \circlearrowright \neg(z \circlearrowright F)) \circlearrowright F && \text{by (SCL4)} \\
&= ((\neg x \circlearrowleft \neg y) \circlearrowleft (z \circlearrowright F)) \circlearrowright F && \text{by (SCL8), (SCL2) and (SCL3)} \\
&= ((\neg x \circlearrowleft (z \circlearrowright F)) \circlearrowright (\neg y \circlearrowleft (z \circlearrowright F))) \circlearrowright F && \text{by (SCL10)} \\
&= (\neg x \circlearrowleft (z \circlearrowright F)) \circlearrowright ((\neg y \circlearrowleft (z \circlearrowright F)) \circlearrowright F) && \text{by (SCL4)} \\
&= (\neg x \circlearrowleft (z \circlearrowright F)) \circlearrowright ((y \circlearrowright \neg(z \circlearrowright F)) \circlearrowright F) && \text{by (SCL8), (SCL2) and (SCL3)} \\
&= ((\neg x \circlearrowleft (z \circlearrowright F)) \circlearrowright y) \circlearrowright (\neg(z \circlearrowright F) \circlearrowright F) && \text{by (SCL4)} \\
&= ((\neg x \circlearrowleft (z \circlearrowright F)) \circlearrowright y) \circlearrowright ((z \circlearrowright F) \circlearrowright F) && \text{by (SCL8)} \\
&= ((\neg x \circlearrowleft (z \circlearrowright F)) \circlearrowright y) \circlearrowright (z \circlearrowright F) && \text{by (SCL4) and (SCL7)} \\
&= (\neg x \circlearrowleft (z \circlearrowright F)) \circlearrowright (y \circlearrowright (z \circlearrowright F)) && \text{by (SCL4)} \\
\\
& (x \circlearrowleft (y \circlearrowright F)) \circlearrowright (z \circlearrowright F) \\
&= (\neg x \circlearrowleft (z \circlearrowright F)) \circlearrowright ((y \circlearrowright F) \circlearrowright (z \circlearrowright F)) && \text{by part (1) of this lemma} \\
&= (\neg x \circlearrowleft (z \circlearrowright F)) \circlearrowright (y \circlearrowright F) && \text{by (SCL7) and (SCL4)} \\
\\
& (x \circlearrowright (y \circlearrowleft T)) \circlearrowleft (z \circlearrowright F) \\
&= (x \circlearrowleft (z \circlearrowright F)) \circlearrowright ((y \circlearrowleft T) \circlearrowleft (z \circlearrowright F)) && \text{by (SCL10)} \\
&= (x \circlearrowleft (z \circlearrowright F)) \circlearrowright (y \circlearrowleft T) && \text{by the duals of (SCL7) and (SCL4) } \square
\end{aligned}$$

Theorem 3.5. *For all $P, Q \in \text{ST}$, if $\text{EqFSCL} \vdash P = Q$ then $\text{FSCL}_{\text{SE}} \models P = Q$.*

Proof. To see that identity, symmetry, transitivity and congruence hold in FSCL_{SE} , we refer the reader to the proof of Theorem 2.5 and note that the proofs for FSCL_{SE} are highly similar.

Verifying the validity of the equations in EqFSCL is cumbersome, but not difficult. As an example we show it for (SCL3). We have

$$\text{SE}(\neg\neg P) = \text{SE}(P)[T \mapsto F, F \mapsto T][T \mapsto F, F \mapsto T] = \text{SE}(P)$$

by a trivial structural induction on evaluation trees. \square

3.1 SCL Normal Form

To aid in our completeness proof we define a normal form for SCL-terms. Because the atoms in SCL-terms may have side effects common normal forms for PL such as Conjunctive Normal Form or Disjunctive Normal Form are not normal forms for SCL. For example, the term $a \circlearrowright (b \circlearrowleft c)$ would be written as $(a \circlearrowright b) \circlearrowleft (a \circlearrowright c)$ in Disjunctive Normal Form, but a trivial examination

shows that the evaluation trees of these terms are not the same. Our normal form is inspired by the FEL Normal Form presented in Chapter 2. We present the grammar for our normal form before we motivate it.

Definition 3.6. *A term $P \in \text{ST}$ is said to be in **SCL Normal Form (SNF)** if it is generated by the following grammar.*

$$\begin{aligned}
P \in \text{SNF} &::= P^\top \mid P^F \mid P^\top \triangleleft P^* \\
P^* &::= P^c \mid P^d \\
P^c &::= P^\ell \mid P^* \triangleleft P^d \\
P^d &::= P^\ell \mid P^* \curlyvee P^c \\
P^\ell &::= (a \triangleleft P^\top) \curlyvee P^F \mid (\neg a \triangleleft P^\top) \curlyvee P^F \\
P^\top &::= \top \mid (a \triangleleft P^\top) \curlyvee P^\top \\
P^F &::= \text{F} \mid (a \curlyvee P^F) \triangleleft P^F,
\end{aligned}$$

where $a \in A$. We refer to P^* -forms as **-terms*, to P^ℓ -forms as *ℓ -terms*, to P^\top -forms as *T-terms* and to P^F -forms as *F-terms*. A term of the form $P^\top \triangleleft P^*$ is referred to as a *T-*-term*.

Without the presence of \top and F in our language, a traditional Negation Normal Form would have sufficed. Furthermore, if $A = \emptyset$, an even more trivial normal form could be used, i.e., just \top or F .

When considering trees in the image of SE we note that some trees only have \top -leaves, some only F -leaves and some both \top -leaves and F -leaves. For any SCL-term P , $\text{SE}(P \curlyvee \top)$ is a tree with only \top -leaves, as can easily be seen from the definition of SE . All terms P such that $\text{SE}(P)$ is a tree with only \top -leaves are rewritten to \top -terms. Similarly, for any term P , $\text{SE}(P \triangleleft \text{F})$ is a tree with only F -leaves. All P such that $\text{SE}(P)$ has only F -leaves are rewritten to F -terms. The simplest trees in the image of SE that have both types of leaves are $\text{SE}(a)$ for $a \in A$. Any (occurrence of an) atom that determines (in whole or in part) the yield of the term, such as a in this example, is referred to as a *determinative* (occurrence of an) atom. This as opposed to a non-determinative (occurrence of an) atom, such as the a in $a \curlyvee \top$, which does not determine (either in whole or in part) the yield of the term. Note that a term P such that $\text{SE}(P)$ contains both \top and F must contain at least one determinative atom.

Terms that contain at least one determinative atom will be rewritten to \top -*-terms. In \top -*-terms we encode each determinative atom together with the non-determinative atoms that occur between it and the next determinative atom in the term (reading from left to right) as an ℓ -term. Observe that the first atom in an ℓ -term is the (only) determinative atom in that ℓ -term and that determinative atoms only occur in ℓ -terms. Also observe that the yield of an ℓ -term is the yield of its determinative atom. This is intuitively convincing, because the remainder of the atoms in any ℓ -term are non-determinative and hence do not contribute to its yield. The non-determinative atoms that may occur before the first determinative atom are encoded as a \top -term. A \top -*-term is the conjunction of a \top -term encoding such atoms and a $*$ -term, which contains only conjunctions and disjunctions of ℓ -terms. We could also have encoded such atoms as an F -term and then taken the disjunction with a $*$ -term to obtain a term with the same semantics. We consider ℓ -terms to be ‘basic’ in $*$ -terms in the sense that they are the smallest grammatical unit that influences the yield of the $*$ -term.

The ℓ -terms in SNF are more complex than those in FEL Normal Form, because short-circuiting allows for the possibility of evaluating different non-determinative atoms depending on the yield of the determinative atom. This is also the reason why the \top -terms and the F -terms

are more complex. Although the atoms occurring in them are not determinative, their yield can influence which atoms in the T-term (F-term) are evaluated next.

We use P^\top , P^* , etc. both to denote grammatical categories and as variables for terms in those categories. The remainder of this section is concerned with defining and proving correct the normalization function $f : \text{ST} \rightarrow \text{SNF}$. We will define f recursively using the functions

$$f^n : \text{SNF} \rightarrow \text{SNF} \quad \text{and} \quad f^c : \text{SNF} \times \text{SNF} \rightarrow \text{SNF}.$$

The first of these will be used to rewrite negated SNF-terms to SNF-terms and the second to rewrite the conjunction of two SNF-terms to an SNF-term. By (SCL2) we have no need for a dedicated function that rewrites the disjunction of two SNF-terms to an SNF-term.

We start by defining f^n . Analyzing the semantics of T-terms and F-terms together with the definition of SE on negations, it becomes clear that f^n must turn T-terms into F-terms and vice versa. We also remark that f^n must preserve the left-associativity of the $*$ -terms in T- $*$ -terms, modulo the associativity within ℓ -terms. We define $f^n : \text{SNF} \rightarrow \text{SNF}$ as follows, using the auxiliary function $f_1^n : P^* \rightarrow P^*$ to ‘push down’ or ‘push in’ the negation symbols when negating a T- $*$ -term. We note that there is no ambiguity between the different grammatical categories present in an SNF-term, i.e., any SNF-term is in exactly one of the grammatical categories identified in Definition 3.6.

$$f^n(\top) = \text{F} \tag{3.1}$$

$$f^n((a \triangleleft P^\top) \circledast Q^\top) = (a \circledast f^n(Q^\top)) \triangleleft f^n(P^\top) \tag{3.2}$$

$$f^n(\text{F}) = \top \tag{3.3}$$

$$f^n((a \circledast P^\text{F}) \triangleleft Q^\text{F}) = (a \triangleleft f^n(Q^\text{F})) \circledast f^n(P^\text{F}) \tag{3.4}$$

$$f^n(P^\top \triangleleft Q^*) = P^\top \triangleleft f_1^n(Q^*) \tag{3.5}$$

$$f_1^n((a \triangleleft P^\top) \circledast Q^\text{F}) = (\neg a \triangleleft f^n(Q^\text{F})) \circledast f^n(P^\top) \tag{3.6}$$

$$f_1^n((\neg a \triangleleft P^\top) \circledast Q^\text{F}) = (a \triangleleft f^n(Q^\text{F})) \circledast f^n(P^\top) \tag{3.7}$$

$$f_1^n(P^* \triangleleft Q^d) = f_1^n(P^*) \circledast f_1^n(Q^d) \tag{3.8}$$

$$f_1^n(P^* \circledast Q^c) = f_1^n(P^*) \triangleleft f_1^n(Q^c). \tag{3.9}$$

Now we turn to defining f^c . These definitions have a great deal of inter-dependence so we first present the definition for f^c when the first argument is a T-term. We see that the conjunction of a T-term with another terms always yields a term of the same grammatical category as the second conjunct.

$$f^c(\top, P) = P \tag{3.10}$$

$$f^c((a \triangleleft P^\top) \circledast Q^\top, R^\top) = (a \triangleleft f^c(P^\top, R^\top)) \circledast f^c(Q^\top, R^\top) \tag{3.11}$$

$$f^c((a \triangleleft P^\top) \circledast Q^\top, R^\text{F}) = (a \circledast f^c(Q^\top, R^\text{F})) \triangleleft f^c(P^\top, R^\text{F}) \tag{3.12}$$

$$f^c((a \triangleleft P^\top) \circledast Q^\top, R^\top \triangleleft S^*) = f^c((a \triangleleft P^\top) \circledast Q^\top, R^\top) \triangleleft S^* \tag{3.13}$$

For defining f^c where the first argument is an F-term, we make use of (SCL7). This immediately shows that the conjunction of an F-term with another term is itself an F-term.

$$f^c(P^\text{F}, Q) = P^\text{F} \tag{3.14}$$

The case where the first conjunct is a T- $*$ -term and the second conjunct is a T-term is defined next. We will use an auxiliary function, $f_1^c : P^* \times P^\top \rightarrow P^*$, to turn conjunctions of a $*$ -term

with a T-term into *-terms. Together with (SCL4) this allows us to define f^c for this case.

$$f^c(P^\top \triangleleft Q^*, R^\top) = P^\top \triangleleft f_1^c(Q^*, R^\top) \quad (3.15)$$

$$f_1^c((a \triangleleft P^\top) \curlyvee Q^F, R^\top) = (a \triangleleft f^c(P^\top, R^\top)) \curlyvee Q^F \quad (3.16)$$

$$f_1^c((\neg a \triangleleft P^\top) \curlyvee Q^F, R^\top) = (\neg a \triangleleft f^c(P^\top, R^\top)) \curlyvee Q^F \quad (3.17)$$

$$f_1^c(P^* \triangleleft Q^d, R^\top) = P^* \triangleleft f_1^c(Q^d, R^\top) \quad (3.18)$$

$$f_1^c(P^* \curlyvee Q^c, R^\top) = f_1^c(P^*, R^\top) \curlyvee f_1^c(Q^c, R^\top) \quad (3.19)$$

When the second conjunct is an F-term, the result will naturally be an F-term itself. So we need to convert the T*-term to an F-term. Using (SCL4) we reduce this problem to converting a *-term to an F-term, for which we use the auxiliary function $f_2^c : P^* \times P^F \rightarrow P^F$.

$$f^c(P^\top \triangleleft Q^*, R^F) = f^c(P^\top, f_2^c(Q^*, R^F)) \quad (3.20)$$

$$f_2^c((a \triangleleft P^\top) \curlyvee Q^F, R^F) = (a \curlyvee Q^F) \triangleleft f^c(P^\top, R^F) \quad (3.21)$$

$$f_2^c((\neg a \triangleleft P^\top) \curlyvee Q^F, R^F) = (a \curlyvee f^c(P^\top, R^F)) \triangleleft Q^F \quad (3.22)$$

$$f_2^c(P^* \triangleleft Q^d, R^F) = f_2^c(P^*, f_2^c(Q^d, R^F)) \quad (3.23)$$

$$f_2^c(P^* \curlyvee Q^c, R^F) = f_2^c(f^n(f_1^c(P^*, f^n(R^F))), f_2^c(Q^c, R^F)) \quad (3.24)$$

Finally we are left with conjunctions of two T*-terms, thus completing the definition of f^c . We use the auxiliary function $f_3^c : P^* \times P^\top \triangleleft P^* \rightarrow P^*$ to ensure that the result is a T*-term.

$$f^c(P^\top \triangleleft Q^*, R^\top \triangleleft S^*) = P^\top \triangleleft f_3^c(Q^*, R^\top \triangleleft S^*) \quad (3.25)$$

$$f_3^c(P^*, Q^\top \triangleleft R^\ell) = f_1^c(P^*, Q^\top) \triangleleft R^\ell \quad (3.26)$$

$$f_3^c(P^*, Q^\top \triangleleft (R^* \triangleleft S^d)) = f_3^c(P^*, Q^\top \triangleleft R^*) \triangleleft S^d \quad (3.27)$$

$$f_3^c(P^*, Q^\top \triangleleft (R^* \curlyvee S^c)) = f_1^c(P^*, Q^\top) \triangleleft (R^* \curlyvee S^c) \quad (3.28)$$

As promised, we now define the normalization function $f : \text{ST} \rightarrow \text{SNF}$ recursively, using f^n and f^c , as follows.

$$f(a) = \top \triangleleft ((a \triangleleft \top) \curlyvee \text{F}) \quad (3.29)$$

$$f(\top) = \top \quad (3.30)$$

$$f(\text{F}) = \text{F} \quad (3.31)$$

$$f(\neg P) = f^n(f(P)) \quad (3.32)$$

$$f(P \triangleleft Q) = f^c(f(P), f(Q)) \quad (3.33)$$

$$f(P \curlyvee Q) = f^n(f^c(f^n(f(P)), f^n(f(Q)))) \quad (3.34)$$

Theorem 3.7. *For any $P \in \text{ST}$, $f(P)$ terminates, $f(P) \in \text{SNF}$ and $\text{EqFSCL} \vdash f(P) = P$.*

In Appendix B.1 we first prove a number of lemmas showing that the definitions f^n and f^c are correct and use those to prove the theorem. We have chosen to use a function rather than a rewriting system to prove the correctness of the normal form, because the author lacks experience with term rewriting systems and because using a function relieves us of the task of proving confluence for the underlying rewriting system.

In Section 4.3 we show that FFEL is a sublogic of FSCL and that any FEL-term can be rewritten to an SCL-term with the same semantics. There we will pay special attention to the application of that translation to terms in FEL Normal Form.

3.2 Tree Structure

In Section 3.3 we will prove that EqFSCl axiomatizes FSCl_{SE} by showing that if $P \in \text{SNF}$ we can invert $\text{SE}(P)$. To do this we need to prove several structural properties of the trees in the image of SE . In the definition of SE we can see how $\text{SE}(P \triangleleft Q)$ is assembled from $\text{SE}(P)$ and $\text{SE}(Q)$ and similarly for $\text{SE}(P \triangleright Q)$. To decompose these trees we will introduce some notation. The trees in the image of SE are all finite binary trees over A with leaves in $\{\mathsf{T}, \mathsf{F}\}$, i.e., $\text{SE}[\text{ST}] \subseteq \mathcal{T}$. We will now also consider the set \mathcal{T}_{\square} of binary trees over A with leaves in $\{\mathsf{T}, \mathsf{F}, \square\}$, where ‘ \square ’ is pronounced ‘box’. The box will be used as a placeholder when composing or decomposing trees. Replacement of the leaves of trees in \mathcal{T}_{\square} by trees in \mathcal{T} or \mathcal{T}_{\square} is defined analogous to replacement for trees in \mathcal{T} , adopting the same notational conventions.

For example we have by definition of SE that $\text{SE}(P \triangleleft Q)$ can be decomposed as

$$\text{SE}(P)[\mathsf{T} \mapsto \square][\square \mapsto \text{SE}(Q)],$$

where $\text{SE}(P)[\mathsf{T} \mapsto \square] \in \mathcal{T}_{\square}$ and $\text{SE}(Q) \in \mathcal{T}$. We note that this only works because the trees in the image of SE , or in \mathcal{T} in general, do not contain any boxes. We start by analyzing the SE -image of ℓ -terms.

Lemma 3.8 (Structure of ℓ -terms). *There is no ℓ -term P such that $\text{SE}(P)$ can be decomposed as $X[\square \mapsto Y]$ with $X \in \mathcal{T}_{\square}$ and $Y \in \mathcal{T}$, where $X \neq \square$, but does contain \square , and Y contains occurrences of both T and F .*

Proof. Let P be some ℓ -term. When we analyze the grammar of P we find that one branch from the root of $\text{SE}(P)$ will only contain T and not F and the other branch vice versa. Hence if $\text{SE}(P) = X[\square \mapsto Y]$ and Y contains occurrences of both T and F , then Y must contain the root and hence $X = \square$. \square

By definition a $*$ -term contains at least one ℓ -term and hence for any $*$ -term P , $\text{SE}(P)$ contains both T and F . The following lemma provides the SE -image of the rightmost ℓ -term in a $*$ -term to witness this fact.

Lemma 3.9 (Determinativeness). *For all $*$ -terms P , $\text{SE}(P)$ can be decomposed as $X[\square \mapsto Y]$ with $X \in \mathcal{T}_{\square}$ and $Y \in \mathcal{T}$ such that X contains \square and $Y = \text{SE}(Q)$ for some ℓ -term Q . Note that X may be \square . We will refer to Y as the witness for this lemma for P .*

Proof. By induction on the complexity of $*$ -terms P modulo the complexity of ℓ -terms. In the base case P is an ℓ -term and $\text{SE}(P) = \square[\square \mapsto \text{SE}(P)]$ is the desired decomposition by Lemma 3.8. For the induction we have to consider both $\text{SE}(P \triangleleft Q)$ and $\text{SE}(P \triangleright Q)$.

We start with $\text{SE}(P \triangleleft Q)$ and let $X[\square \mapsto Y]$ be the decomposition for $\text{SE}(Q)$ which we have by induction hypothesis. Since by definition of SE on \triangleleft we have

$$\text{SE}(P \triangleleft Q) = \text{SE}(P)[\mathsf{T} \mapsto \text{SE}(Q)]$$

we also have

$$\text{SE}(P \triangleleft Q) = \text{SE}(P)[\mathsf{T} \mapsto X[\square \mapsto Y]] = \text{SE}(P)[\mathsf{T} \mapsto X][\square \mapsto Y].$$

The last equality is due to the fact that $\text{SE}(P)$ does not contain any boxes. This gives our desired decomposition. The case for $\text{SE}(P \triangleright Q)$ is analogous. \square

The following lemma illustrates another structural property of trees in the image of $*$ -terms under SE , namely that the left branch of any determinative atom in such a tree is different from its right branch.

Lemma 3.10 (Non-decomposition). *There is no $*$ -term P such that $\text{SE}(P)$ can be decomposed as $X[\Box \mapsto Y]$ with $X \in \mathcal{T}_\Box$ and $Y \in \mathcal{T}$, where $X \neq \Box$ and X contains \Box , but not \top or \bot .*

Proof. By induction on P modulo the complexity of ℓ -terms. The base case covers ℓ -terms and follows immediately from Lemma 3.9 ($\text{SE}(P)$ contains occurrences of both \top and \bot) and Lemma 3.8 (no non-trivial decomposition exists that contains both). For the induction we assume that the lemma holds for all $*$ -terms with lesser complexity than $P \frown Q$ and $P \vee Q$.

We start with the case for $\text{SE}(P \frown Q)$. Suppose for contradiction that $\text{SE}(P \frown Q) = X[\Box \mapsto Y]$ with $X \neq \Box$ and X not containing any occurrences of \top or \bot . Let R be a witness of Lemma 3.9 for P . Now note that $\text{SE}(P \frown Q)$ has a subtree $R[\top \mapsto \text{SE}(Q)]$. Because Y must contain both the occurrences of \bot in the one branch of $R[\top \mapsto \text{SE}(Q)]$ as well as the occurrences of $\text{SE}(Q)$ in the other (because they contain \top and \bot), Lemma 3.8 implies that Y must (strictly) contain $\text{SE}(Q)$. Hence there is a $Z \in \mathcal{T}$ such that $\text{FE}(P) = X[\Box \mapsto Z]$, which violates the induction hypothesis. The case for $\text{SE}(P \vee Q)$ is symmetric. \square

We now arrive at two crucial definitions for our completeness proof. When considering $*$ -terms, we already know that $\text{SE}(P \frown Q)$ can be decomposed as

$$\text{SE}(P)[\top \mapsto \Box][\Box \mapsto \text{SE}(Q)].$$

Our goal now is to give a definition for a type of decomposition so that this is the only such decomposition for $\text{SE}(P \frown Q)$. We also ensure that $\text{SE}(P \vee Q)$ does not have a decomposition of that type, so that we can distinguish $\text{SE}(P \frown Q)$ from $\text{SE}(P \vee Q)$. Similarly, we need to define another type of decomposition so that $\text{SE}(P \vee Q)$ can only be decomposed as

$$\text{SE}(P)[\bot \mapsto \Box][\Box \mapsto \text{SE}(Q)]$$

and that $\text{SE}(P \frown Q)$ does not have a decomposition of that type.

Definition 3.11. *The pair $(Y, Z) \in \mathcal{T}_\Box \times \mathcal{T}$ is a **candidate conjunction decomposition (ccd)** of $X \in \mathcal{T}$, if*

- $X = Y[\Box \mapsto Z]$,
- Y contains \Box ,
- Y contains \bot , but not \top , and
- Z contains both \top and \bot .

*Similarly, (Y, Z) is a **candidate disjunction decomposition (cdd)** of X , if*

- $X = Y[\Box \mapsto Z]$,
- Y contains \Box ,
- Y contains \top , but not \bot , and
- Z contains both \top and \bot .

We note that the ccd and cdd are not necessarily the decompositions we are looking for, because, for example, $\text{SE}((P \frown Q) \frown R)$ has a ccd $(\text{SE}(P)[\top \mapsto \Box], \text{SE}(Q \frown R))$, whereas the decomposition we need is $(\text{SE}(P \frown Q)[\top \mapsto \Box], \text{SE}(R))$. Therefore we refine these definitions to obtain the decompositions we seek.

Definition 3.12. *The pair $(Y, Z) \in \mathcal{T}_\Box \times \mathcal{T}$ is a **conjunction decomposition (cd)** of $X \in \mathcal{T}$, if it is a ccd of X and there is no other ccd (Y', Z') of X where the depth of Z' is smaller than that of Z . Similarly, (Y, Z) is a **disjunction decomposition (dd)** of X , if it is a cdd of X and there is no other cdd (Y', Z') of X where the depth of Z' is smaller than that of Z .*

Theorem 3.13. For any $*$ -term $P \triangleleft Q$, i.e., with $P \in P^*$ and $Q \in P^d$, $\text{SE}(P \triangleleft Q)$ has the (unique) cd

$$(\text{SE}(P)[\top \mapsto \square], \text{SE}(Q))$$

and no dd. For any $*$ -term $P \vee Q$, i.e., with $P \in P^*$ and $Q \in P^c$, $\text{SE}(P \vee Q)$ has no cd and its (unique) dd is

$$(\text{SE}(P)[\text{F} \mapsto \square], \text{SE}(Q)).$$

Proof. By simultaneous induction on $P \triangleleft Q$ and $P \vee Q$ modulo the complexity of ℓ -terms. In the basis we have to consider, for ℓ -terms P and Q , the terms $P \triangleleft Q$ and $P \vee Q$. Both of these are covered by the cases in the induction where the second conjunct (or disjunct) Q is an ℓ -term. This is valid reasoning, since we don't call upon the induction hypothesis in those cases. For the induction we assume that the theorem holds for all $*$ -terms with lesser complexity than $P \triangleleft Q$ and $P \vee Q$. We first treat the case for $P \triangleleft Q$.

First for the cd. Note that $(\text{SE}(P)[\top \mapsto \square], \text{SE}(Q))$ is a ccd of $\text{SE}(P \triangleleft Q)$ by definition of SE on \triangleleft (for the first condition) and Lemma 3.9 (for the third and fourth condition). We also know that for any ccd (Y, Z) either Z contains or is contained in $\text{SE}(Q)$. For suppose otherwise, then Y will contain an occurrence of \top , namely the one we know by Lemma 3.9 that $\text{SE}(Q)$ has. By the above it suffices to show that there is no ccd (Y, Z) where Z is strictly contained in $\text{SE}(Q)$. Suppose for contradiction that such a ccd (Y, Z) does exist.

By definition of $*$ -terms Q is either an ℓ -term or a disjunction. If Q is an ℓ -term and Z is strictly contained in $\text{SE}(Q)$ then Z does not contain both \top and F by Lemma 3.8. Therefore $(\text{SE}(P)[\top \mapsto \square], \text{SE}(Q))$ is the *unique* cd for $\text{SE}(P \triangleleft Q)$.

If Q is a disjunction, then if Z is strictly contained in $\text{SE}(Q)$ we can decompose $\text{SE}(Q)$ as $\text{SE}(Q) = U[\square \mapsto Z]$ for some $U \in \mathcal{T}_\square$ that contains but is not equal to \square . By Lemma 3.10 this implies that U contains either \top or F . If it contains \top , then so does Y , because

$$Y = \text{SE}(P)[\top \mapsto U],$$

and (Y, Z) is not a ccd for $\text{SE}(P \triangleleft Q)$. If it only contains F then (U, Z) is a ccd for $\text{SE}(Q)$ which violates the induction hypothesis. Therefore $(\text{SE}(P)[\top \mapsto \square], \text{SE}(Q))$ is the *unique* cd for $\text{SE}(P \triangleleft Q)$.

Now for the dd. It suffices to show that there is no cdd for $\text{SE}(P \triangleleft Q)$. Again Q is either an ℓ -term or a disjunction. Suppose for contradiction that (Y, Z) is a cdd for $\text{SE}(P \triangleleft Q)$. If Q is an ℓ -term, then Z must contain all occurrences of F in $\text{SE}(P \triangleleft Q)$. So in particular it must contain all occurrences of F in $\text{SE}(Q)$. It must also contain at least one occurrence of \top . Hence by Lemma 3.8 Z must contain $\text{SE}(Q)$. But then Z contains all the occurrences of \top in $\text{SE}(P \triangleleft Q)$ and hence Y does not contain any occurrences of \top . Therefore there is no cdd for $\text{SE}(P \triangleleft Q)$.

If Q is a disjunction then Z must contain all occurrences of F in $\text{SE}(P \triangleleft Q)$. Let R be a witness of Lemma 3.9 for P . Now note that $R[\top \mapsto \text{SE}(Q)]$ is a subtree of $\text{SE}(P \triangleleft Q)$. Also note that Lemma 3.8 implies that there is no way to decompose $R[\top \mapsto \text{SE}(Q)]$ such that $R[\top \mapsto \text{SE}(Q)] = U[\square \mapsto V]$ for some $U \in \mathcal{T}_\square$ that contains but is not equal to \square and some $V \in \mathcal{T}$ containing occurrences of both $\text{SE}(Q)$ and F . So because Z must contain all occurrences of F in $\text{SE}(P \triangleleft Q)$, it must strictly contain $\text{SE}(Q)$. But all the occurrences of \top in $\text{SE}(P \triangleleft Q)$ are in occurrences of $\text{SE}(Q)$. Hence Y does not contain any occurrences of \top . Therefore there is no cdd for $\text{SE}(P \triangleleft Q)$. The case for $\text{SE}(P \vee Q)$ is symmetric. \square

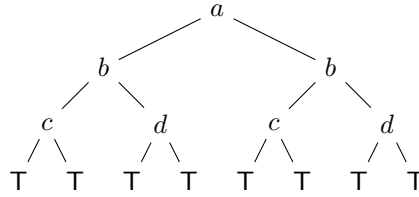
At this point we have the tools necessary to invert SE on $*$ -terms, at least down to the level of ℓ -terms. We can easily detect if a tree in the image of SE is in the image of P^ℓ , because all leaves to the left of the root are one truth value, while all the leaves to the right are the other. To invert SE on \top - $*$ -terms we still need to be able to reconstruct $\text{SE}(P^\top)$ and $\text{SE}(Q^*)$ from

$\text{SE}(P^\top \triangleleft Q^*)$. To this end we define a \top -*-decomposition, as with cds and dds we first define a candidate \top -*-decomposition.

Definition 3.14. *The pair $(Y, Z) \in \mathcal{T}_\square \times \mathcal{T}$ is a **candidate \top -*-decomposition (ctsd)** of $X \in \mathcal{T}$, if $X = Y[\square \mapsto Z]$, Y does not contain \top or F and there is no decomposition $(U, V) \in \mathcal{T}_\square \times \mathcal{T}$ of Z such that*

- $Z = U[\square \mapsto V]$,
- U contains \square ,
- $U \neq \square$, and
- U contains neither \top nor F .

Unlike with FEL, this is not the decomposition we seek in this case. Consider for example that there is a \top -term with the following semantics:



Let P^\top be the \top -term with these semantics and observe that $\text{SE}(P^\top \triangleleft Q^*)$ has a ctsd

$$(\square \triangleleft a \triangleleft \square, (\text{SE}(Q^*) \triangleleft c \triangleleft \text{SE}(Q^*)) \triangleleft b \triangleleft (\text{SE}(Q^*) \triangleleft d \triangleleft \text{SE}(Q^*))).$$

But the decomposition we seek is $(\text{SE}(P^\top)[\top \mapsto \square], \text{SE}(Q^*))$. Hence we will refine this definition to aid in the theorem below.

Definition 3.15. *The pair $(Y, Z) \in \mathcal{T}_\square \times \mathcal{T}$ is a **\top -*-decomposition (tsd)** of $X \in \mathcal{T}$, if it is a ctsd of X and there is no other ctsd (Y', Z') of X where the depth of Z' is smaller than that of Z .*

Theorem 3.16. *For any \top -term P and $*$ -term Q the (unique) tsd of $\text{SE}(P \triangleleft Q)$ is*

$$(\text{SE}(P)[\top \mapsto \square], \text{SE}(Q)).$$

Proof. First we observe that $(\text{SE}(P)[\top \mapsto \square], \text{SE}(Q))$ is a ctsd because by definition of SE on \triangleleft we have $\text{SE}(P)[\top \mapsto \text{SE}(Q)] = \text{SE}(P \triangleleft Q)$ and $\text{SE}(Q)$ is non-decomposable by Lemma 3.10.

Suppose for contradiction that there is ctsd (Y, Z) such that the depth of Z is smaller than that of $\text{SE}(Q)$. Now Z must contain or be contained in $\text{SE}(Q)$ for otherwise Y would contain \top or F , i.e., the ones we know $\text{SE}(Q)$ has by Lemma 3.9. Clearly the former cannot be the case, for then Z would have a greater depth than $\text{SE}(Q)$. So the latter is the case and $\text{SE}(Q) = U[\square \mapsto Z]$ for some $U \in \mathcal{T}_\square$ that is not equal to \square and does not contain \top or F (because then Y would too). But this violates Lemma 3.10, which states that no such decomposition exists. \square

3.3 Completeness

With the two theorems from the previous section, we can prove completeness for FSCL_{SE} . We define three auxiliary functions to aid in our definition of the inverse of SE on SNF . Let $\text{cd} : \mathcal{T} \rightarrow \mathcal{T}_\square \times \mathcal{T}$ be the function that returns the conjunction decomposition of its argument, dd of the

same type its disjunction decomposition and tsd , also of the same type, its $\text{T-}*$ -decomposition. Naturally, these functions are undefined when their argument does not have a decomposition of the specified type. Each of these functions returns a pair and we will use cd_1 (dd_1 , tsd_1) to denote the first element of this pair and cd_2 (dd_2 , tsd_2) to denote the second element.

We define $g : \mathcal{T} \rightarrow \text{ST}$ using the functions $g^\top : \mathcal{T} \rightarrow \text{ST}$ for inverting trees in the image of T -terms and g^F , g^ℓ and g^* of the same type for inverting trees in the image of F -terms, ℓ -terms and $*$ -terms, respectively. These functions are defined as follows.

$$g^\top(X) = \begin{cases} \top & \text{if } X = \top \\ (a \delta \wedge g^\top(Y)) \vee g^\top(Z) & \text{if } X = Y \trianglelefteq a \trianglerighteq Z \end{cases} \quad (3.35)$$

$$g^\text{F}(X) = \begin{cases} \text{F} & \text{if } X = \text{F} \\ (a \vee g^\text{F}(Z)) \delta \wedge g^\text{F}(Y) & \text{if } X = Y \trianglelefteq a \trianglerighteq Z \end{cases} \quad (3.36)$$

$$g^\ell(X) = \begin{cases} (a \delta \wedge g^\top(Y)) \vee g^\text{F}(Z) & \text{if } X = Y \trianglelefteq a \trianglerighteq Z \text{ for some } a \in A \\ & \text{and } Y \text{ only has } \top\text{-leaves} \\ (\neg a \delta \wedge g^\top(Z)) \vee g^\text{F}(Y) & \text{if } X = Y \trianglelefteq a \trianglerighteq Z \text{ for some } a \in A \\ & \text{and } Z \text{ only has } \top\text{-leaves} \end{cases} \quad (3.37)$$

$$g^*(X) = \begin{cases} g^*(\text{cd}_1(X)[\square \mapsto \top]) \delta \wedge g^*(\text{cd}_2(X)) & \text{if } X \text{ has a cd} \\ g^*(\text{dd}_1(X)[\square \mapsto \text{F}]) \vee g^*(\text{dd}_2(X)) & \text{if } X \text{ has a dd} \\ g^\ell(X) & \text{otherwise} \end{cases} \quad (3.38)$$

$$g(X) = \begin{cases} g^\top(X) & \text{if } X \text{ has only } \top\text{-leaves} \\ g^\text{F}(X) & \text{if } X \text{ has only } \text{F}\text{-leaves} \\ g^\top(\text{tsd}_1(X)[\square \mapsto \top]) \delta \wedge g_*(\text{tsd}_2(X)) & \text{otherwise} \end{cases} \quad (3.39)$$

Theorem 3.17. *For all $P \in \text{SNF}$, $g(\text{SE}(P)) \equiv P$.*

The proof for this theorem can be found in Appendix B.2. For the sake of completeness, we separately state the completeness result below.

Theorem 3.18. *For all $P, Q \in \text{ST}$, if $\text{FSCL}_{\text{SE}} \models P = Q$ then $\text{EqFSCL} \vdash P = Q$.*

Proof. It suffices to show that for $P, Q \in \text{SNF}$, $\text{SE}(P) = \text{SE}(Q)$ implies $P \equiv Q$. To see this suppose that P' and Q' are two SCL -terms and $\text{SE}(P') = \text{SE}(Q')$. We know that P' is derivably equal to an SNF -term P , i.e., $\text{EqFSCL} \vdash P' = P$, and that Q' is derivably equal to an SNF -term Q , i.e., $\text{EqFSCL} \vdash Q' = Q$. Theorem 3.5 then gives us $\text{SE}(P') = \text{SE}(P)$ and $\text{SE}(Q') = \text{SE}(Q)$. Hence by the result $P \equiv Q$ and in particular $\text{EqFSCL} \vdash P = Q$. Transitivity then gives us $\text{EqFSCL} \vdash P' = Q'$ as desired.

The result follows immediately from Theorem 3.17. \square

In Section 4.2 we use this result to prove that FSCL_{SE} is equivalent to FSCL as it is defined by Bergstra and Ponse in [BP10b].

Relating FFEL to FSCL and Proposition Algebra

This chapter is concerned with explaining the connection between FFEL and FSCL, a connection we formalize in the setting of Proposition Algebra. Because the main results of this thesis were not presented in this setting, we will forgo a detailed introduction to Proposition Algebra. Instead we refer the reader to [BP11] for a proper introduction to Proposition Algebra and to [BP10b] and [BP12] for an introduction to FSCL as it is defined in terms of Proposition Algebra. We will however, very briefly, fix some core concepts and notation from this setting.

In [BP11] Bergstra and Ponse introduce Proposition Algebra for reasoning about the sequential evaluation of propositional terms using the ternary connective $y \triangleleft x \triangleright z$, to be read as ‘if x then y else z ’ and called ‘Hoare’s conditional operator’, defined in [Hoa85]. The terms under consideration can be described in Backus-Naur Form, letting A be a countable set of atoms, by

$$P \in \text{CT} ::= a \in A \mid \top \mid \text{F} \mid P \triangleleft P \triangleright P.$$

The equality of two of these terms is defined by the set of axioms CP, for ‘Conditional Propositions’:

$$x \triangleleft \top \triangleright y = x \tag{CP1}$$

$$x \triangleleft \text{F} \triangleright y = y \tag{CP2}$$

$$\top \triangleleft x \triangleright \text{F} = x \tag{CP3}$$

$$x \triangleleft (y \triangleleft z \triangleright u) \triangleright v = (x \triangleleft y \triangleright v) \triangleleft z \triangleright (x \triangleleft u \triangleright v). \tag{CP4}$$

When the equality of two terms s and t in CT, possibly containing variables, can be derived from equational logic and (CP1)–(CP4), we denote this by $\text{CP} \vdash s = t$. Bergstra and Ponse extend Proposition Algebra with negation and the short-circuit connectives \triangleleft and \triangleright to obtain the set CT_s of closed terms, where we see that $\text{ST} \subset \text{CT}_s$. They extend CP with the following defining equations for the newly introduced connectives.

$$\neg x = \text{F} \triangleleft x \triangleright \top \tag{4.1}$$

$$x \triangleleft y = y \triangleleft x \triangleright \text{F} \tag{4.2}$$

$$x \triangleright y = \top \triangleleft x \triangleright y \tag{4.3}$$

Let CP_s denote CP together with these defining equations. If the equality of two terms s and t in CT_s , possibly containing variables, can be derived from equational logic and CP_s , we denote this by $\text{CP}_s \vdash s = t$. Free Short-Circuit Logic (FSCL) is then defined as follows. For all $P, Q \in \text{ST}$,

$$\text{FSCL} \models P = Q \iff \text{CP}_s \vdash P = Q.$$

We will show in Section 4.1 that FFEL can also be expressed in terms of Proposition Algebra with an extended signature, by adding defining equations for the *full* left-sequential connectives to CP. In Section 4.2 we will prove that FSCL_{SE} , as we defined it in Chapter 3, is equivalent to FSCL. Finally in Section 4.3 we will prove that FFEL is a sublogic of FSCL by showing that its connectives are definable in FSCL, which will allow us to define a general left-sequential logic.

4.1 Relating FFEL to Proposition Algebra

To relate FFEL to Proposition Algebra, we first add negation, \blacktriangleleft and \blacktriangleright to the signature of Proposition Algebra, to obtain the set CT_f of closed terms, where we note that $\text{FT} \subset \text{CT}_f$. We then extend the set of equations CP with the following defining equations for the full left-sequential connectives.

$$\neg x = \text{F} \blacktriangleleft x \blacktriangleright \text{T} \tag{4.4}$$

$$x \blacktriangleleft y = y \blacktriangleleft x \blacktriangleright (\text{F} \blacktriangleleft y \blacktriangleright \text{F}) \tag{4.5}$$

$$x \blacktriangleright y = (\text{T} \blacktriangleleft y \blacktriangleright \text{T}) \blacktriangleleft x \blacktriangleright y \tag{4.6}$$

Let CP_f denote CP together with these three equations. When two terms $s, t \in \text{CT}_f$, possibly containing variables, are derivable by equational logic and CP_f we denote this by $\text{CP}_f \vdash s = t$.

Our goal is to prove that FFEL can also be characterized by CP_f , i.e., we will show that for all $P, Q \in \text{FT}$,

$$\text{FFEL} \models P = Q \iff \text{CP}_f \vdash P = Q.$$

To this end we first define the function $\text{CE} : \text{CT} \rightarrow \mathcal{T}$ that will interpret CT-terms in \mathcal{T} . We will then extend this definition to $\text{CE}_f : \text{CT}_f \rightarrow \mathcal{T}$.

With the informal semantics of ‘if x then y else z ’ in mind for terms of the form $y \blacktriangleleft x \blacktriangleright z$, defining CE becomes straightforward.

Definition 4.1. *Let A be a countable set of atoms and let \mathcal{T} be the set all finite binary trees over A with leaves in $\{\text{T}, \text{F}\}$. We define the unary **Conditional Evaluation** function $\text{CE} : \text{CT} \rightarrow \mathcal{T}$ as:*

$$\begin{aligned} \text{CE}(\text{T}) &= \text{T} \\ \text{CE}(\text{F}) &= \text{F} \\ \text{CE}(a) &= \text{T} \trianglelefteq a \trianglerighteq \text{F} && \text{for } a \in A \\ \text{CE}(Q \blacktriangleleft P \blacktriangleright R) &= \text{CE}(P)[\text{T} \mapsto \text{CE}(Q), \text{F} \mapsto \text{CE}(R)] \end{aligned}$$

We observe that $\text{CE}[\text{CT}] = \mathcal{T}$. We extend this definition to $\text{CE}_f : \text{CT}_f \rightarrow \mathcal{T}$ by adding the following clauses to the definition.

$$\begin{aligned} \text{CE}_f(\neg P) &= \text{CE}_f(P)[\text{T} \mapsto \text{F}, \text{F} \mapsto \text{T}] \\ \text{CE}_f(P \blacktriangleleft Q) &= \text{CE}_f(P)[\text{T} \mapsto \text{CE}_f(Q), \text{F} \mapsto \text{CE}_f(Q)[\text{T} \mapsto \text{F}]] \\ \text{CE}_f(P \blacktriangleright Q) &= \text{CE}_f(P)[\text{T} \mapsto \text{CE}_f(Q)[\text{F} \mapsto \text{T}], \text{F} \mapsto \text{CE}_f(Q)] \end{aligned}$$

It is now trivial to see that CE_f restricted to FEL-terms is equal to FE.

Theorem 4.2. *For all $P, Q \in \text{FT}$,*

$$\text{FFEL} \models P = Q \iff \text{CP}_f \vdash P = Q.$$

Proof. First we show that $\text{FFEL} \models P = Q$ implies that $\text{CP}_f \vdash P = Q$. If $\text{FFEL} \models P = Q$, then $\text{FE}(P) = \text{FE}(Q)$ and by Theorem 2.17 $\text{EqFFEL} \vdash P = Q$. It suffices to prove that EqFFEL is sound with respect to CP_f . It is immediate that identity, symmetry, transitivity and congruence are sound and checking that (FEL1)–(FEL10) are valid is cumbersome, but not difficult. As an example we show this for (FEL8):

$$\begin{aligned} P \blacklozenge F &= F \triangleleft P \triangleright (F \triangleleft F \triangleright F) && \text{by (4.5)} \\ &= (F \triangleleft F \triangleright (F \triangleleft F \triangleright F)) \triangleleft P \triangleright F && \text{by (CP2)} \\ &= (F \triangleleft F \triangleright (F \triangleleft F \triangleright F)) \triangleleft P \triangleright (F \triangleleft T \triangleright (F \triangleleft F \triangleright F)) && \text{by (CP1)} \\ &= F \triangleleft (F \triangleleft P \triangleright T) \triangleright (F \triangleleft F \triangleright F) && \text{by (CP4)} \\ &= \neg P \blacklozenge F && \text{by (4.5) and (4.4)} \end{aligned}$$

Given this soundness, $\text{FFEL} \models P = Q$ implies $\text{CP}_f \vdash P = Q$.

Next we show that $\text{CP}_f \vdash P = Q$ implies $\text{FFEL} \models P = Q$. Because CE_f restricted to FT equals FE , it suffices to show that CP_f is sound with respect to CE_f -equality, i.e., that for all $R, S \in \text{CT}_f$, $\text{CP}_f \vdash R = S$ implies $\text{CE}_f(R) = \text{CE}_f(S)$. This proof is also trivial. As an example we show this for (CP2) as follows:

$$\begin{aligned} \text{CE}_f(R \triangleleft F \triangleright S) &= \text{CE}_f(F)[T \mapsto \text{CE}_f(R), F \mapsto \text{CE}_f(S)] \\ &= F[T \mapsto \text{CE}_f(R), F \mapsto \text{CE}_f(S)] \\ &= \text{CE}_f(S), \end{aligned}$$

and for (4.5) as

$$\begin{aligned} \text{CE}_f(R \blacklozenge S) &= \text{CE}_f(R)[T \mapsto \text{CE}_f(S), F \mapsto \text{CE}_f(S)[T \mapsto F]] \\ &= \text{CE}_f(R)[T \mapsto \text{CE}_f(S), F \mapsto \text{CE}_f(S)[T \mapsto F, F \mapsto F]] \\ &= \text{CE}_f(R)[T \mapsto \text{CE}_f(S), F \mapsto \text{CE}_f(F \triangleleft S \triangleright F)] \\ &= \text{CE}_f(S \triangleleft R \triangleright (F \triangleleft S \triangleright F)). \end{aligned} \quad \square$$

Given the definition of FSCL , the reason for defining FFEL in terms of evaluation trees rather than by using Proposition Algebra deserves some clarification. We feel that this makes our completeness proof more straightforward than it would have been had we defined FFEL in terms of Proposition Algebra. Although CT -terms are easily interpreted as trees, we would have had to use a basic form for CT -terms, such as [BP11, Definition 3.1], to perform our analysis as done in Section 2.2. In fact our function CE converts CT -terms to such basic forms if we read ‘ \triangleleft ’ as ‘ \triangleleft ’ and ‘ \triangleright ’ as ‘ \triangleright ’, thus explaining our choice of notation for trees. We have taken the notation $y \triangleleft x \triangleright z$ from the setting of Thread Algebra, see, e.g., [PVdZ06], where it is used to denote the post-conditional composition of threads.

4.2 EqFSCL axiomatizes FSCL

It has been an open question since FSCL was first defined in [BP10b] whether or not EqFSCL , or an equivalent set of equations such as the one presented in [BP10b] itself, axiomatizes FSCL . Given Theorem 3.18 it now suffices to show that FSCL_{SE} is equivalent to FSCL , which we shall prove analogous to how we proved the theorem in the previous section.

We define the function $\text{CE}_s : \text{CT}_s \rightarrow \mathcal{T}$ that interprets CT_s -terms as evaluation trees by extending the definition of CE with the following clauses.

$$\begin{aligned}\text{CE}_s(\neg P) &= \text{CE}_s(P)[\mathsf{T} \mapsto \mathsf{F}, \mathsf{F} \mapsto \mathsf{T}] \\ \text{CE}_s(P \mathbin{\mathcal{J}} Q) &= \text{CE}_s(P)[\mathsf{T} \mapsto \text{CE}_s(Q)] \\ \text{CE}_s(P \mathbin{\mathcal{V}} Q) &= \text{CE}_s(P)[\mathsf{F} \mapsto \text{CE}_s(Q)]\end{aligned}$$

It is now trivial to see that CE_s restricted to SCL -terms is equal to SE .

Theorem 4.3. *For all $P, Q \in \text{ST}$,*

$$\text{FSCL}_{\text{SE}} \models P = Q \iff \text{FSCL} \models P = Q.$$

Proof. If $\text{FSCL}_{\text{SE}} \models P = Q$, then by Theorem 3.18 we have $\text{EqFSCL} \vdash P = Q$. So it suffices to show that EqFSCL is sound with respect to CP_s , i.e., that $\text{EqFSCL} \vdash P = Q$ implies $\text{CP}_s \vdash P = Q$. This proof can be found in [BP10b].

For the other direction we must show that if $\text{CP}_s \vdash P = Q$, then $\text{FSCL}_{\text{SE}} \models P = Q$. Because CE_s restricted to ST equals SE , it suffices to show that CP_s is sound with respect to CE_s -equality, i.e., that for all $R, S \in \text{CT}_s$, $\text{CP}_s \vdash R = S$ implies $\text{CE}_s(R) = \text{CE}_s(S)$. This is again a trivial proof. For example, we show it for $(\text{CP}3)$ as follows:

$$\begin{aligned}\text{CE}_s(\mathsf{T} \triangleleft R \triangleright \mathsf{F}) &= \text{CE}_s(R)[\mathsf{T} \mapsto \text{CE}_s(\mathsf{T}), \mathsf{F} \mapsto \text{CE}_s(\mathsf{F})] \\ &= \text{CE}_s(R)[\mathsf{T} \mapsto \mathsf{T}, \mathsf{F} \mapsto \mathsf{F}] \\ &= \text{CE}_s(R),\end{aligned}$$

and for (4.3) as

$$\begin{aligned}\text{CE}_s(R \mathbin{\mathcal{V}} S) &= \text{CE}_s(R)[\mathsf{F} \mapsto \text{CE}_s(S)] \\ &= \text{CE}_s(R)[\mathsf{T} \mapsto \mathsf{T}, \mathsf{F} \mapsto \text{CE}_s(S)] \\ &= \text{CE}_s(\mathsf{T} \triangleleft R \triangleright S).\end{aligned}$$

□

The reader may wonder why in this thesis we presented the completeness of EqFSCL with respect to FSCL_{SE} rather than FSCL . The reason for this is that the author discovered the result after proving the completeness of EqFFEL with respect to FFEL , and this presentation emphasizes the similarities and differences of that proof with the proof for FSCL .

4.3 FFEL is a sublogic of FSCL

When we consider a simple FEL -term such as $a \mathbin{\mathcal{J}} b$ and picture $\text{FE}(a \mathbin{\mathcal{J}} b)$, we see that we can reconstruct the normal form of the original term as $\mathsf{T} \mathbin{\mathcal{J}} ((a \mathbin{\mathcal{J}} \mathsf{T}) \mathbin{\mathcal{J}} (b \mathbin{\mathcal{J}} \mathsf{T}))$. However, we can also reconstruct this tree as $\mathsf{T} \mathbin{\mathcal{J}} ((a \mathbin{\mathcal{J}} \mathsf{T}) \mathbin{\mathcal{V}} ((b \mathbin{\mathcal{V}} \mathsf{F}) \mathbin{\mathcal{J}} \mathsf{F})) \mathbin{\mathcal{J}} ((b \mathbin{\mathcal{J}} \mathsf{T}) \mathbin{\mathcal{V}} \mathsf{F})$. We will indeed show that for any FEL -term P there is an SCL -term Q such that $\text{FE}(P) = \text{SE}(Q)$. To this end we define a translation function h , which translates FEL -terms to SCL -terms with the same evaluation tree semantics as follows.

$$h(\mathsf{T}) = \mathsf{T} \tag{4.7}$$

$$h(\mathsf{F}) = \mathsf{F} \tag{4.8}$$

$$h(a) = a \tag{4.9}$$

$$h(\neg P) = \neg h(P) \tag{4.10}$$

$$h(P \mathbin{\mathcal{J}} Q) = (h(P) \mathbin{\mathcal{V}} (h(Q) \mathbin{\mathcal{J}} \mathsf{F})) \mathbin{\mathcal{J}} h(Q) \tag{4.11}$$

$$h(P \mathbin{\mathcal{V}} Q) = (h(P) \mathbin{\mathcal{J}} (h(Q) \mathbin{\mathcal{V}} \mathsf{T})) \mathbin{\mathcal{V}} h(Q) \tag{4.12}$$

We immediately turn to the proof that h has the desired property.

Theorem 4.4. *FFEL is a sublogic of FSCL, i.e., for all $P, Q \in FT$,*

$$\text{FFEL} \models P = Q \implies \text{FSCL} \models h(P) = h(Q).$$

Proof. It suffices by Theorem 4.3 to show that for any FEL-term P , $\text{FE}(P) = \text{SE}(h(P))$, which we shall do by induction on P . The base cases are trivial. For the induction we have

$$\begin{aligned} \text{FE}(P \blacktriangleleft Q) &= \text{FE}(P)[\mathsf{T} \mapsto \text{FE}(Q), \mathsf{F} \mapsto \text{FE}(Q)[\mathsf{T} \mapsto \mathsf{F}]] \\ &= \text{SE}(h(P))[\mathsf{T} \mapsto \text{SE}(h(Q)), \mathsf{F} \mapsto \text{SE}(h(Q))[\mathsf{T} \mapsto \mathsf{F}]] \quad \text{by induction hypothesis} \\ &= \text{SE}(h(P))[\mathsf{F} \mapsto \text{SE}(h(Q))[\mathsf{T} \mapsto \mathsf{F}]][\mathsf{T} \mapsto \text{SE}(h(Q))] \\ &= \text{SE}((h(P) \circlearrowleft (h(Q) \blacktriangleleft \mathsf{F})) \blacktriangleleft h(Q)) \\ &= \text{SE}(h(P \blacktriangleleft Q)), \end{aligned}$$

where the third equality follows from the fact that $\text{SE}(h(Q))[\mathsf{T} \mapsto \mathsf{F}]$ does not contain T . The case for $P \blacktriangleright Q$ is similar. \square

Let us consider the translation of FEL-terms in P^\top . In the base case we have that $h(\mathsf{T}) = \mathsf{T}$ and in the inductive case we have $h(a \blacktriangleright P) = (a \blacktriangleleft (h(P) \circlearrowleft \mathsf{T})) \circlearrowleft h(P)$. By Lemma B.1 and the induction hypothesis this is equal to $(a \blacktriangleleft h(P)) \circlearrowleft h(P)$. In other words, FEL-terms in P^\top are equivalent to SCL-terms in P^\top . Similarly, FEL-terms in P^F are translated to SCL-terms in P^F . In both cases they are equivalent with T -terms (F -terms) P for which the paths of $\text{SE}(P)$ all contain the same atoms *in the same order*.

In Chapter 1 we promised to define a general left-sequential logic, i.e., a logic for reasoning about propositional terms that contain both short-circuit left-sequential connectives and full left-sequential connectives. We can now easily define such a logic by adding the following two equations to EqFSCL:

$$x \blacktriangleleft y = (x \circlearrowleft (y \blacktriangleleft \mathsf{F})) \blacktriangleleft y \quad \text{and} \quad x \blacktriangleright y = (x \blacktriangleleft (y \circlearrowleft \mathsf{T})) \circlearrowleft y.$$

By the results from Chapter 3 and this chapter, it is immediate that this set of equations axiomatizes a (free) general left-sequential logic. Naturally, we could also express this logic in terms of Proposition Algebra by adding both types of connectives to its signature and considering CP together with (4.1)–(4.3), (4.5) and (4.6). Without making any assumptions about the side effects that atoms may have, this logic can be used to reason about propositional terms in programming languages which offer both types of connectives, such as Java.

In the next chapter we will discuss our motivations for examining FEL separately from SCL.

Conclusion and Outlook

The evaluation strategy prescribed by a propositional-based logic is key to determining the semantics of propositional terms as they are used in programming languages to direct the flow of a program. For any such evaluation strategy to be of use to programmers, it must be deterministic. For suppose otherwise, then two evaluations of the same term in the same execution environment could yield different results and such a system can hardly be called a logic. Given that these evaluation strategies must be deterministic, we are left with considering sequential evaluation strategies and parallel evaluation strategies.

We have not found any programming language that uses a parallel evaluation strategy when dealing with propositional terms for program flow control. The likely reason being that to truly evaluate several subterms in parallel, the state of the entire execution environment prior to the start of the evaluation must be copied so that each subterm can be evaluated in the same environment. This would likely cause the evaluation to be slowed down to such an extent as to render it useless in practice. The ‘merging’ of multiple copies of the environment after the subterms have been evaluated would also be a highly non-trivial exercise. Perhaps in the setting of Quantum Computing we can imagine an evaluation in superposition resulting in a superposition of environments, but it is not at all clear how we should interpret the superposition of the yields of the individual atoms.

Therefore we focus on sequential evaluation strategies, which are widely used in programming languages. We focus entirely on left-sequential evaluation strategies, since most programming languages read from left to right. We have examined both short-circuit evaluation strategies, in the form of FSCL_{SE} , and full evaluation strategies, in the form of FFEL . Both induce right-sequential evaluation strategies. For example, for FEL we could introduce the symbols ‘ \wedge_\bullet ’ for full right-sequential conjunction and ‘ \vee_\bullet ’ for full right-sequential disjunction. The equations in EqFFEL could then easily be rewritten to accommodate the new direction, e.g., (FEL9) would become $y \vee_\bullet (F \wedge_\bullet x) = y \wedge_\bullet (T \vee_\bullet x)$. For examining a setting with both left-sequential and right-sequential connectives, we would naturally define the right-sequential connectives in terms of their left-sequential counterparts.

In ordinary propositional logic no particular evaluation strategy is prescribed. SAT-solvers make eager use of this freedom and often employ complex evaluation strategies that go far beyond simple left-sequential evaluation. See, e.g., [GPFW96] for a survey of some of the different algorithms used for satisfiability solving. We emphasize that SAT-solvers deal with propositional terms whose atoms do not have side effects. Both FFEL and FSCL_{SE} are designed to deal with atoms that *do* have side effects, to which can be contributed much of the complexity of these

logics.

As promised in Chapter 1 we return to our claim about the applicability of left-sequential logics for reasoning about side effects. The yield of any occurrence of an atom $a \in A$ in some term P can be influenced by the side effects of the atoms that precede the occurrence of a in P as well as the state of the execution environment in which P is evaluated. If an atom $a \in A$ does not have a side effect, then it always behaves either as the constant \mathbf{T} or as the constant \mathbf{F} depending on the atoms that were evaluated before it and the state of the execution environment. For $a \in A$ and $P \in \text{ST}$ let $[T/a]P$ denote the term which results from replacing each occurrence of a in P by \mathbf{T} . Similarly, let $[F/a]P$ be the term that results from replacing each occurrence of a in P with \mathbf{F} . Let y_e be the function that returns the boolean yield of an SCL-term when it is evaluated in execution environment e . An atom $a \in A$ has a side effect if there is some execution environment e and there are $P, Q \in \text{ST}$ with $y_e(P) = y_e(Q)$ such that either

$$y_e([T/a]P) \neq y_e([T/a]Q) \quad \text{or} \quad y_e([F/a]P) \neq y_e([F/a]Q).$$

As an example consider atoms a and b and suppose that a side effect of a is that any evaluation of b that follows it will yield \mathbf{T} . Also suppose that if b were not preceded by a it would yield \mathbf{F} . To make this concrete we could imagine a being a method that sets some global variable in the execution environment and always yields \mathbf{T} . We could then see b as being a method that checks whether that variable has been set, in which case it yields \mathbf{T} , or not, in which case it yields \mathbf{F} . Letting e be some execution environment where the global variable is not set, or alternatively the empty execution environment, we see that $y_e(a \blacktriangle b) = \mathbf{T} = y_e(\neg b)$ and that $y_e(\mathbf{T} \blacktriangle b) = \mathbf{F} \neq y_e(\neg b)$. Hence a has a side effect by our definition.

This opens the door to more involved reasoning about side effects. The example above hints at the possibility of defining what it means for an atom to be ‘impacted’ by the side effect of a single other atom. We could, for example, restrict our attention to $P, Q \in \text{ST}$ containing only a and some other atom b . If there is an environment e and there are $P, Q \in \text{ST}$ that contain only a and b such that $y_e(P) = y_e(Q)$, but $y_e([T/a]P) \neq y_e([T/a]Q)$ or $y_e([F/a]P) \neq y_e([F/a]Q)$, then we know that b is impacted by a side effect of a . Another interesting definition would be that of a ‘positive side effect’. In that case we could say that an atom a has a positive side effect if there is some environment e and some $P, Q \in \text{ST}$ such that $y_e(P) = y_e(Q)$, but $y_e([T/a]P) \neq y_e([T/a]Q)$.

With this definition of a side effect we see that FEL, unlike PL, preserves side effects in the sense that $\text{FFEL} \models P = Q$ implies $\text{FFEL} \models [T/a]P = [T/a]Q$ and $\text{FFEL} \models [F/a]P = [F/a]Q$ for all $P, Q \in \text{FT}$ and all $a \in A$. The same goes for SCL. Thus, if we adopt our proposed definition of side effects, both FEL and SCL can be used to reason about propositional expressions with atoms that may have side effects.

For defining and reasoning about side effects as we have done above we need constants for our truth values. The constants \mathbf{T} and \mathbf{F} are not definable in SCL and FEL, except in terms of one another. This is unlike PL, where the law of excluded middle allows one to define \mathbf{T} in terms of negation and disjunction, naturally assuming we have at least one atom at our disposal. The constant \mathbf{F} can then be defined in terms of \mathbf{T} or by the law of non-contradiction. With SCL and FEL we cannot exclude the possibility that every atom has a side effect that causes any subsequent evaluation of that same atom to yield the opposite truth value. Hence neither the law of excluded middle nor the law of non-contradiction are valid in SCL or FEL.

We now turn to the question of the usefulness of FEL. As mentioned in Chapter 1 we find that some programming languages offer full left-sequential connectives, which motivates the initial investigation of FEL. We claim that FEL has a greater value than merely to act as means of writing certain SCL terms using fewer symbols. The usefulness of a full evaluation strategy lies in the increased predictability of the state of the environment after a (sub)term has been

evaluated. In particular, we know that the side effects of all the atoms in the term have occurred. Naturally we leave errors and error values out of this discussion.

SCL is characterized by its efficiency, in the sense that atoms are not evaluated if their yield is not needed to determine the yield of a term as a whole. From that perspective FEL might seem rather inefficient, but this is not necessarily so. To determine the state of the environment after the evaluation of a FEL term in a given environment, we need only compute how each atom in the term transforms the environment. It is *not* necessary to compute the yield of any of the atoms. With SCL terms in general we must know what the first atom yields in order to determine which atom is next to transform the environment. Thus to compute the state of the environment after the evaluation of an SCL term we must compute the yield of each atom that transforms the environment *and* we must compute the transformation of the environment for each atom that affects it. Consider the SCL-term

$$((a \vee (b \wedge F)) \wedge F) \vee c$$

and note that to compute its yield we must first compute the yield of a to determine whether or not b is short-circuited. Consider atoms a and b that have no side effects, and hence do not affect the environment. If computing the yield of a is computationally very demanding, the FEL-term

$$((a \bullet \vee (b \bullet \wedge F)) \bullet \wedge F) \bullet \vee c$$

can be evaluated more quickly, because it is not necessary to compute the yield of a .

In [BP10b] several variants of SCL are defined in addition to FSCL. In this thesis we have only defined one variant of FEL, i.e., FFEL. An important variant of SCL is Memorizing Short-Circuit Logic, MSCL, which is defined in the same way as FSCL, but adding the following axiom to CP:

$$x \triangleleft y \triangleright (z \triangleleft u \triangleright (v \triangleleft y \triangleright w)) = x \triangleleft y \triangleright (z \triangleleft u \triangleright w) \quad (\text{CPmem})$$

As we can see from (CPmem), once an atom has been evaluated all subsequent evaluations of the same atom will yield the same truth value. An example of such a ‘memorizing’ atom in programming might be a call to a memoizing function¹ with a fixed argument. Naturally we could define Memorizing Fully Evaluated Left-Sequential Logic, MFEL, in a similar fashion. Given our evaluation tree semantics however, we can also define a ‘post-processing’ on our trees instead. We simply take the FE image of a term and recursively walk down the tree. Whenever we encounter $X \triangleleft a \triangleright Y$ in a left subtree of an a , we replace it by $X \triangleleft a \triangleright X$. Similarly, we replace it by $Y \triangleleft a \triangleright Y$ if we are in the right subtree of an a .

Another variant of SCL is Static SCL, SSCL, which is defined in [BP10b] in the same way as FSCL, but adding (CPmem) and the following equation to CP.

$$F \triangleleft x \triangleright F = F$$

This equation implies that $x \wedge F = F$, and more generally, $x \wedge y = y \wedge x$. As shown in [BP10b], this variant is the same as PL, except that a particular evaluation strategy is prescribed. Naturally we could define Static FEL, SFEL, similarly. We believe that the most elegant method of defining variants, other than the free variants, for FEL and SCL is by means of Proposition Algebra. We believe that evaluation trees offer a didactically interesting alternative definition for the free variants, because they offer a straightforward semantics just for the left-sequential connectives.

When considering these and other variants of FEL and SCL it is useful to consider what these logics express in terms of the properties of atoms. Any atom in MFEL (or MSCL) is memorizing

¹A memoizing function is a function which maintains a cache of function values for arguments it has previously been called with. See, e.g., <http://en.wikipedia.org/wiki/Memoization> for a detailed description of memoization.

in the sense that its yield becomes constant after it is first evaluated. The atoms in SFEL (or SSCL) have no side effects according to our definition. For practical applications of the theory of left-sequential logics it may be useful to partition the set A of atoms into sets of atoms possessing certain of these properties. A compound logic, geared towards the optimization of propositional statements in programming, could then be defined. In such a logic, for example, we would have $x \triangleleft F = F$ for any atom in the ‘static’ partition of A . The potential for optimization, i.e., evaluating as few atoms as possible to compute the yield of a term, becomes even greater when we consider the variants of Contractive SCL and Repetition-Proof SCL, see [BP10b].

In [Reg10] Regenboog showed that CP is ω -complete if and only if, for A a countable set of atoms, $|A| > 1$. For an axiomatization of Static CP which is interderivable with the one presented above he showed ω -completeness for any countable set of atoms. He also showed that the axioms in CP and several axiomatizations extending it are independent. We have shown neither ω -completeness nor independence for EqFFEL, although we would consider such theorems valuable future work. It is also an open question whether EqFSCL is ω -complete or independent. The set EqFSCL as presented in [BP10b] and [BP12] is a different set from the one we have introduced in Chapter 3, although they are interderivable. The set we presented came about in consultation with the authors of the original definition. Because the set EqFSCL is somewhat ‘in flux’ in this sense, and to a lesser degree because its independence and ω -completeness are still open questions, we have refrained from referring to the equations in EqFSCL as axioms. Similarly, our definition of EqFFEL differs slightly from that in [Blo11], hence we do not refer to those equations as axioms either. Formal definitions of variants of FEL other than FFEL and a comparative analysis of these and the corresponding variants of SCL we also consider a great avenue for further study.

Acknowledgments

Firstly I would like to thank my thesis supervisor, Alban Ponse, for the many hours he spent reviewing drafts and providing valuable feedback. His experience has proved a great help in cleaning up the notation used in this work as well as its presentation in general. If it were not for his confidence in an earlier version of the completeness proof for FEL, I might never have discovered the completeness proof for SCL and improved the presentation of the FEL proof significantly in the process.

I also wish to thank Alwin Blok for the work he has done on Fully Evaluated Left-Sequential Logic, in particular his discovery of the axiomatization for Free FEL and an early version of a FEL Normal Form, together with its proof of correctness. Our discussions about the motivations for that normal form and the semantic structures of evaluation trees have been of great help.

Finally, I thank my family for the moral support they gave me while writing this thesis.

Proofs for FFEL

A.1 Correctness of f

To prove that $f : \text{FT} \rightarrow \text{FNF}$ is indeed a normalization function we need to prove that for all FEL-terms P , $f(P)$ terminates, $f(P) \in \text{FNF}$ and $\text{EqFFEL} \vdash f(P) = P$. To arrive at this result, we prove several intermediate results about the functions f^n and f^c , roughly in the order in which their definitions were presented in Section 2.1. For the sake of brevity we will not explicitly prove that these functions terminate. To see that each function terminates consider that a termination proof would closely mimic the proof structure of the lemmas dealing with the grammatical categories of the images of these functions.

Lemma A.1. *For any P^F and P^\top , $\text{EqFFEL} \vdash P^F = P^F \blacktriangleleft F$ and $\text{EqFFEL} \vdash P^\top = P^\top \blacktriangleright \top$.*

Proof. We prove both claims simultaneously by induction. In the base case we have $F = \top \blacktriangleleft F$ by (FEL5), which is equal to $F \blacktriangleleft F$ by (FEL8) and (FEL1). The base case for the second claim follows from that for the first claim by duality.

For the induction we have $a \blacktriangleleft P^F = a \blacktriangleleft (P^F \blacktriangleleft F)$ by the induction hypothesis and the result follows from (FEL4). For the second claim we again appeal to duality. \square

Lemma A.2. *The following equations can be derived by equational logic and EqFFEL.*

$$1. \ x \blacktriangleleft (y \blacktriangleleft (z \blacktriangleleft F)) = (x \blacktriangleright y) \blacktriangleleft (z \blacktriangleleft F)$$

$$2. \ \neg x \blacktriangleleft (y \blacktriangleright \top) = \neg(x \blacktriangleleft (y \blacktriangleright \top))$$

Proof.

$$\begin{aligned}
x \blacktriangleleft (y \blacktriangleleft (z \blacktriangleleft F)) &= x \blacktriangleleft ((\neg y \blacktriangleleft z) \blacktriangleleft F) && \text{by (FEL4) and 2.4 (1)} \\
&= (\neg x \blacktriangleleft \neg y) \blacktriangleleft (z \blacktriangleleft F) && \text{by (FEL4) and 2.4 (1)} \\
&= \neg(\neg x \blacktriangleleft \neg y) \blacktriangleleft (z \blacktriangleleft F) && \text{by Lemma 2.4 (1)} \\
&= (x \blacktriangleright y) \blacktriangleleft (z \blacktriangleleft F) && \text{by (FEL2)} \\
\neg x \blacktriangleleft (y \blacktriangleright \top) &= \neg x \blacktriangleright (y \blacktriangleleft F) && \text{by (FEL10)} \\
&= \neg(x \blacktriangleleft \neg(y \blacktriangleleft F)) && \text{by (FEL2) and (FEL3)} \\
&= \neg(x \blacktriangleleft \neg(\neg y \blacktriangleleft \neg \top)) && \text{by (FEL8) and (FEL1)} \\
&= \neg(x \blacktriangleleft (y \blacktriangleright \top)) && \text{by (FEL2)} \quad \square
\end{aligned}$$

Lemma A.3. *For all $P \in \text{FNF}$, if P is a \top -term then $f^n(P)$ is an F -term, if it is an F -term then $f^n(P)$ is a \top -term, if it is a \top -*-term then so is $f^n(P)$, and*

$$\text{EqFFEL} \vdash f^n(P) = \neg P.$$

Proof. We start with proving the claims for \top -terms, by induction on P^\top . In the base case $f^n(\top) = \text{F}$. It is immediate that $f^n(\top)$ is an F -term. The claim that $\text{EqFFEL} \vdash f^n(\top) = \neg \top$ is immediate by (FEL1). For the inductive case we have that $f^n(a \blacktriangleright P^\top) = a \blacktriangleleft f^n(P^\top)$, where we may assume that $f^n(P^\top)$ is an F -term and that $\text{EqFFEL} \vdash f^n(P^\top) = \neg P^\top$. The grammatical claim now follows immediately from the induction hypothesis. Furthermore, noting that by the induction hypothesis we may assume that $f^n(P^\top)$ is an F -term, we have:

$$\begin{aligned} f^n(a \blacktriangleright P^\top) &= a \blacktriangleleft f^n(P^\top) && \text{by definition} \\ &= a \blacktriangleleft (f^n(P^\top) \blacktriangleleft \text{F}) && \text{by Lemma A.1} \\ &= \neg a \blacktriangleleft (f^n(P^\top) \blacktriangleleft \text{F}) && \text{by Lemma 2.4 (1)} \\ &= \neg a \blacktriangleleft f^n(P^\top) && \text{by Lemma A.1} \\ &= \neg a \blacktriangleleft \neg P^\top && \text{by induction hypothesis} \\ &= \neg(a \blacktriangleright P^\top). && \text{by (FEL3) and (FEL2)} \end{aligned}$$

For F -terms we prove our claims by induction on P^F . In the base case $f^n(\text{F}) = \top$. It is immediate that $f^n(\text{F})$ is a \top -term. The claim that $\text{EqFFEL} \vdash f^n(\text{F}) = \neg \text{F}$ is immediate by the dual of (FEL1). For the inductive case we have that $f^n(a \blacktriangleleft P^\text{F}) = a \blacktriangleright f^n(P^\text{F})$, where we may assume that $f^n(P^\text{F})$ is a \top -term and $\text{EqFFEL} \vdash f^n(P^\text{F}) = \neg P^\text{F}$. It follows immediately from the induction hypothesis that $f^n(a \blacktriangleleft P^\text{F})$ is a \top -term. Furthermore, noting that by the induction hypothesis we may assume that $f^n(P^\text{F})$ is a \top -term, we prove the remaining claim as follows:

$$\begin{aligned} f^n(a \blacktriangleleft P^\text{F}) &= a \blacktriangleright f^n(P^\text{F}) && \text{by definition} \\ &= a \blacktriangleright (f^n(P^\text{F}) \blacktriangleright \top) && \text{by Lemma A.1} \\ &= \neg a \blacktriangleright (f^n(P^\text{F}) \blacktriangleright \top) && \text{by the dual of Lemma 2.4 (1)} \\ &= \neg a \blacktriangleright f^n(P^\text{F}) && \text{by Lemma A.1} \\ &= \neg a \blacktriangleright \neg P^\text{F} && \text{by induction hypothesis} \\ &= \neg(a \blacktriangleleft P^\text{F}). && \text{by (FEL3) and (FEL2)} \end{aligned}$$

To prove the lemma for \top -*-terms we first verify that the auxiliary function f_1^n returns a \top -*-term and that for any \top -*-term P , $\text{EqFFEL} \vdash f_1^n(P) = \neg P$. We show this by induction on the number of ℓ -terms in P . For the base cases, i.e., for ℓ -terms, it is immediate that $f_1^n(P)$ is a \top -*-term. If P is an ℓ -term with a positive determinative atom we have:

$$\begin{aligned} f_1^n(a \blacktriangleleft P^\top) &= \neg a \blacktriangleleft P^\top && \text{by definition} \\ &= \neg a \blacktriangleleft (P^\top \blacktriangleright \top) && \text{by Lemma A.1} \\ &= \neg(a \blacktriangleleft (P^\top \blacktriangleright \top)) && \text{by Lemma A.2 (2)} \\ &= \neg(a \blacktriangleleft P^\top). && \text{by Lemma A.1} \end{aligned}$$

If P is an ℓ -term with a negative determinative atom the proof proceeds the same, substituting $\neg a$ for a and applying (FEL3) where needed. For the inductive step we assume that the result holds for \top -terms with fewer ℓ -terms than $P^* \blacktriangleleft Q^d$ and $P^* \blacktriangleright Q^c$. We note that each application of

f_1^n changes the main connective (not occurring inside an ℓ -term) and hence the result is a $*$ -term. Derivable equality is, given the induction hypothesis, an instance of (the dual of) (FEL2).

With this result we can now see that $f^n(P^\top \blacktriangle Q^*)$ is indeed a \top - $*$ -term. Furthermore we find that:

$$\begin{aligned}
f^n(P^\top \blacktriangle Q^*) &= P^\top \blacktriangle f_1^n(Q^*) && \text{by definition} \\
&= P^\top \blacktriangle \neg Q^* && \text{as shown above} \\
&= (P^\top \blacktriangledown \top) \blacktriangle \neg Q^* && \text{by Lemma A.1} \\
&= \neg(P^\top \blacktriangledown \top) \blacktriangledown \neg Q^* && \text{by Lemma 2.4 (2)} \\
&= \neg P^\top \blacktriangledown \neg Q^* && \text{by Lemma A.1} \\
&= \neg(P^\top \blacktriangle Q^*). && \text{by (FEL2) and (FEL3)}
\end{aligned}$$

Hence for all $P \in \text{FNF}$, $\text{EqFFEL} \vdash f^n(P) = \neg P$. \square

Lemma A.4. *For any \top -term P and $Q \in \text{FNF}$, $f^c(P, Q)$ has the same grammatical category as Q and*

$$\text{EqFFEL} \vdash f^c(P, Q) = P \blacktriangle Q.$$

Proof. By induction on the complexity of the first argument. In the base case we see that $f^c(\top, P) = P$ and hence has the same grammatical category as P . Derivable equality follows from (FEL5).

For the induction step we make a case distinction on the grammatical category of the second argument. If the second argument is a \top -term we have that $f^c(a \blacktriangledown P^\top, Q^\top) = a \blacktriangledown f^c(P^\top, Q^\top)$, where we assume that $f^c(P^\top, Q^\top)$ is a \top -term and $\text{EqFFEL} \vdash f^c(P^\top, Q^\top) = P^\top \blacktriangle Q^\top$. The grammatical claim follows immediately from the induction hypothesis. The claim about derivable equality is proved as follows:

$$\begin{aligned}
f^c(a \blacktriangledown P^\top, Q^\top) &= a \blacktriangledown f^c(P^\top, Q^\top) && \text{by definition} \\
&= a \blacktriangledown (P^\top \blacktriangle Q^\top) && \text{by induction hypothesis} \\
&= a \blacktriangledown (P^\top \blacktriangle (Q^\top \blacktriangledown \top)) && \text{by Lemma A.1} \\
&= (a \blacktriangledown P^\top) \blacktriangle (Q^\top \blacktriangledown \top) && \text{by Lemma 2.4 (3)} \\
&= (a \blacktriangledown P^\top) \blacktriangle Q^\top. && \text{by Lemma A.1}
\end{aligned}$$

If the second argument is an F -term we assume that $f^c(P^\top, Q^F)$ is an F -term and that $\text{EqFFEL} \vdash f^c(P^\top, Q^F) = P^\top \blacktriangle Q^F$. The grammatical claim follows immediately from the induction hypothesis. Derivable equality is proved as follows:

$$\begin{aligned}
f^c(a \blacktriangledown P^\top, Q^F) &= a \blacktriangle f^c(P^\top, Q^F) && \text{by definition} \\
&= a \blacktriangle (P^\top \blacktriangle Q^F) && \text{by induction hypothesis} \\
&= a \blacktriangle (P^\top \blacktriangle (Q^F \blacktriangle F)) && \text{by Lemma A.1} \\
&= (a \blacktriangledown P^\top) \blacktriangle (Q^F \blacktriangle F) && \text{by Lemma A.2 (1)} \\
&= (a \blacktriangledown P^\top) \blacktriangle Q^F. && \text{by Lemma A.1}
\end{aligned}$$

Finally, if the second argument is a \top - $*$ -term then $f^c(a \blacktriangledown P^\top, Q^\top \blacktriangle R^*) = f^c(a \blacktriangledown P^\top, Q^\top) \blacktriangle R^*$. The fact that this is a \top - $*$ -term follows from the fact that $f^c(a \blacktriangledown P^\top, Q^\top)$ is a \top -term as was shown above. Derivable equality follows from the case where the second argument is a \top -term and (FEL4). \square

Lemma A.5. *For any T -*-term P and F -term Q , $f^c(P, Q)$ is an F -term and*

$$\text{EqFFEL} \vdash f^c(P, Q) = P \blacktriangleleft Q.$$

Proof. By (FEL4) and Lemma A.4 it suffices to show that $f_2^c(P^*, Q^F)$ is an F -term and that $\text{EqFFEL} \vdash f_2^c(P^*, Q^F) = P^* \blacktriangleleft Q^F$. We prove this by induction on the number of ℓ -terms in P^* . In the base cases, i.e., ℓ -terms, the grammatical claims follow from Lemma A.4. The claim about derivable equality in the case of ℓ -terms with positive determinative atoms follows from Lemma A.4 and (FEL4). For ℓ -terms with negative determinative atoms it follows from Lemma A.4, Lemma A.1, (FEL7), (FEL4) and (FEL8).

For the induction step we assume the claims hold for any T -terms with fewer ℓ -terms than $P^* \blacktriangleleft Q^d$ and $P^* \blacktriangleright Q^c$. In the case of conjunctions we have $f_2^c(P^* \blacktriangleleft Q^d, R^F) = f_2^c(P^*, f_2^c(Q^d, R^F))$ and the grammatical claim follows from the induction hypothesis (applied twice). Derivable equality follows from the induction hypothesis and (FEL4).

For disjunctions we have $f_2^c(P^* \blacktriangleright Q^c, R^F) = f_2^c(P^*, f_2^c(Q^c, R^F))$ and the grammatical claim follows from the induction hypothesis (applied twice). The claim about derivable equality is proved as follows:

$$\begin{aligned} f_2^c(P^* \blacktriangleright Q^c, R^F) &= f_2^c(P^*, f_2^c(Q^c, R^F)) && \text{by definition} \\ &= P^* \blacktriangleleft (Q^c \blacktriangleleft R^F) && \text{by induction hypothesis} \\ &= P^* \blacktriangleleft (Q^c \blacktriangleleft (R^F \blacktriangleleft F)) && \text{by Lemma A.1} \\ &= (P^* \blacktriangleright Q^c) \blacktriangleleft (R^F \blacktriangleleft F) && \text{by Lemma A.2 (1)} \\ &= (P^* \blacktriangleright Q^c) \blacktriangleleft R^F. && \text{by Lemma A.1} \quad \square \end{aligned}$$

Lemma A.6. *For any F -term P and $Q \in \text{FNF}$, $f^c(P, Q)$ is an F -term and*

$$\text{EqFFEL} \vdash f^c(P, Q) = P \blacktriangleleft Q.$$

Proof. We make a case distinction on the grammatical category of the second argument. If the second argument is a T -term we proceed by induction on the first argument. In the base case we have $f^c(F, P^\top) = f^n(P^\top)$ and the result is by Lemma A.3, Lemma A.1, (FEL7) and (FEL8). In the inductive case we have $f^c(a \blacktriangleleft P^F, Q^\top) = a \blacktriangleleft f^c(P^F, Q^\top)$, where we assume that $f^c(P^F, Q^\top)$ is an F -term and $\text{EqFFEL} \vdash f^c(P^F, Q^\top) = P^F \blacktriangleleft Q^\top$. The result now follows from the induction hypothesis and (FEL4).

If the second argument is an F -term the proof is almost the same, except that we need not invoke Lemma A.3 or (FEL8) in the base case.

Finally, if the second argument is a T -*-term we again proceed by induction on the first argument. In the base case we have $f^c(F, P^\top \blacktriangleleft Q^*) = f^c(P^\top \blacktriangleleft Q^*, F)$. The grammatical claim now follows from Lemma A.5 and derivable equality follows from Lemma A.5 and (FEL7). For the inductive case the results follow from the induction hypothesis and (FEL4). \square

Lemma A.7. *For any T -*-term P and T -term Q , $f^c(P, Q)$ has the same grammatical category as P and*

$$\text{EqFFEL} \vdash f^c(P, Q) = P \blacktriangleleft Q.$$

Proof. By (FEL4) it suffices to prove the claims for f_1^c , i.e., that $f_1^c(P^*, Q^\top)$ has the same grammatical category as P^* and that $\text{EqFFEL} \vdash f_1^c(P^*, Q^\top) = P^* \blacktriangleleft Q^\top$. We prove this by induction on the number of ℓ -terms in P^* . In the base case we deal with ℓ -terms and the results follow from Lemma A.4 and (FEL4).

For the inductive cases we assume that the results hold for any $*$ -term with fewer ℓ -terms than $P^* \blacktriangleleft Q^d$ and $P^* \blacktriangledown Q^c$. In the case of conjunctions the results follow from the induction hypothesis and (FEL4). In the case of disjunctions the grammatical claim follows from the induction hypothesis. For derivable equality we have:

$$\begin{aligned}
f_1^c(P^* \blacktriangledown Q^c, R^\top) &= P^* \blacktriangledown f_1^c(Q^c, R^\top) && \text{by definition} \\
&= P^* \blacktriangledown (Q^c \blacktriangleleft R^\top) && \text{by induction hypothesis} \\
&= P^* \blacktriangledown (Q^c \blacktriangleleft (R^\top \blacktriangledown \top)) && \text{by Lemma A.1} \\
&= (P^* \blacktriangledown Q^c) \blacktriangleleft (R^\top \blacktriangledown \top) && \text{by Lemma 2.4 (3)} \\
&= (P^* \blacktriangledown Q^c) \blacktriangleleft R^\top. && \text{by Lemma A.1} \quad \square
\end{aligned}$$

Lemma A.8. *For any $P, Q \in \text{FNF}$, $f^c(P, Q)$ is in FNF and*

$$\text{EqFFEL} \vdash f^c(P, Q) = P \blacktriangleleft Q.$$

Proof. By the four preceding lemmas it suffices to show that $f^c(P^\top \blacktriangleleft Q^*, R^\top \blacktriangleleft S^*)$ is in FNF and that $\text{EqFFEL} \vdash f^c(P^\top \blacktriangleleft Q^*, R^\top \blacktriangleleft S^*) = (P^\top \blacktriangleleft Q^*) \blacktriangleleft (R^\top \blacktriangleleft S^*)$. By (FEL4), in turn, it suffices to prove that $f_3^c(P^*, Q^\top \blacktriangleleft R^*)$ is a $*$ -term and that $\text{EqFFEL} \vdash f_3^c(P^*, Q^\top \blacktriangleleft R^*) = P^* \blacktriangleleft (Q^\top \blacktriangleleft R^*)$. We prove this by induction on the number of ℓ -terms in R^* . In the base case we have that $f_3^c(P^*, Q^\top \blacktriangleleft R^\ell) = f_1^c(P^*, Q^\top) \blacktriangleleft R^\ell$. The results follow from Lemma A.7 and (FEL4).

For the inductive cases we assume that the results hold for all $*$ -terms with fewer ℓ -terms than $R^* \blacktriangleleft S^d$ and $R^* \blacktriangledown S^c$. For conjunctions the result follows from the induction hypothesis and (FEL4) and for disjunctions it follows from Lemma A.7 and (FEL4). \square

Theorem 2.7. *For any $P \in \text{FT}$, $f(P)$ terminates, $f(P) \in \text{FNF}$ and $\text{EqFFEL} \vdash f(P) = P$.*

Proof. By induction on the complexity of P . If P is an atom, the result is by (FEL5) and (FEL6). If P is \top or F the result is by identity. For the induction we assume that the result holds for all FEL-terms of lesser complexity than $P \blacktriangleleft Q$ and $P \blacktriangledown Q$. The result now follows from the induction hypothesis, Lemma A.3, Lemma A.8 and (FEL2). \square

A.2 Correctness of g

Theorem 2.16. *For all $P \in \text{FNF}$, $g(\text{FE}(P)) \equiv P$.*

Proof. We first prove that for all \top -terms P , $g^\top(\text{FE}(P)) \equiv P$, by induction on P . In the base case $P \equiv \top$ and we have $g^\top(\text{FE}(P)) \equiv g^\top(\top) \equiv \top \equiv P$. For the inductive case we have $P \equiv a \blacktriangledown Q^\top$ and

$$\begin{aligned}
g^\top(\text{FE}(P)) &\equiv g^\top(\text{FE}(Q^\top) \triangleleft a \triangleright \text{FE}(Q^\top)) && \text{by definition of FE} \\
&\equiv a \blacktriangledown g^\top(\text{FE}(Q^\top)) && \text{by definition of } g^\top \\
&\equiv a \blacktriangledown Q^\top && \text{by induction hypothesis} \\
&\equiv P.
\end{aligned}$$

Similarly, we see that for all F -terms P , $g^\text{F}(\text{FE}(P)) \equiv P$, by induction on P . In the base case $P \equiv \text{F}$ and we have $g^\text{F}(\text{FE}(P)) \equiv g^\text{F}(\text{F}) \equiv \text{F} \equiv P$. For the inductive case we have $P \equiv a \blacktriangleleft Q^\text{F}$

and

$$\begin{aligned}
g^F(\text{FE}(P)) &\equiv g^F(\text{FE}(Q^F) \leq a \geq \text{FE}(Q^F)) && \text{by definition of FE} \\
&\equiv a \bullet g^F(\text{FE}(Q^F)) && \text{by definition of } g^F \\
&\equiv a \bullet Q^F && \text{by induction hypothesis} \\
&\equiv P.
\end{aligned}$$

Now we check that for all ℓ -terms P , $g^\ell(\text{FE}(P)) \equiv P$. We observe that either $P \equiv a \bullet Q^\top$ or $P \equiv \neg a \bullet Q^\top$. In the first case we have

$$\begin{aligned}
g^\ell(\text{FE}(P)) &\equiv g^\ell(\text{FE}(Q^\top) \leq a \geq \text{FE}(Q^\top)[\top \mapsto \text{F}]) && \text{by definition of FE} \\
&\equiv a \bullet g^\ell(\text{FE}(Q^\top)) && \text{by definition of } g^\ell \\
&\equiv a \bullet Q^\top && \text{as shown above} \\
&\equiv P.
\end{aligned}$$

In the second case we have that

$$\begin{aligned}
g^\ell(\text{FE}(P)) &\equiv g^\ell(\text{FE}(Q^\top)[\top \mapsto \text{F}] \leq a \geq \text{FE}(Q^\top)) && \text{by definition of FE} \\
&\equiv \neg a \bullet g^\ell(\text{FE}(Q^\top)) && \text{by definition of } g^\ell \\
&\equiv \neg a \bullet Q^\top && \text{as shown above} \\
&\equiv P.
\end{aligned}$$

We now prove that for all $*$ -terms P , $g^*(\text{FE}(P)) \equiv P$, by induction on P modulo the complexity of ℓ -terms. In the base case we are dealing with ℓ -terms. Because an ℓ -term has neither a cd nor a dd we have $g^*(\text{FE}(P)) \equiv g^\ell(\text{FE}(P)) \equiv P$, where the first equality is by definition of g^* and the second was shown above. For the induction we have either $P \equiv Q \bullet R$ or $P \equiv Q \blacktriangleright R$. In the first case note that by Theorem 2.13, $\text{FE}(P)$ has a cd and no dd. So we have

$$\begin{aligned}
g^*(\text{FE}(P)) &\equiv g^*(\text{cd}_1(\text{FE}(P))[\square_1 \mapsto \top, \square_2 \mapsto \text{F}]) \bullet g^*(\text{cd}_2(\text{FE}(P))) && \text{by definition of } g^* \\
&\equiv g^*(\text{FE}(Q)) \bullet g^*(\text{FE}(R)) && \text{by Theorem 2.13} \\
&\equiv Q \bullet R && \text{by induction hypothesis} \\
&\equiv P.
\end{aligned}$$

In the second case, again by Theorem 2.13, $\text{FE}(P)$ has a dd and no cd. So we have that

$$\begin{aligned}
g^*(\text{FE}(P)) &\equiv g^*(\text{dd}_1(\text{FE}(P))[\square_1 \mapsto \top, \square_2 \mapsto \text{F}]) \blacktriangleright g^*(\text{dd}_2(\text{FE}(P))) && \text{by definition of } g^* \\
&\equiv g^*(\text{FE}(Q)) \blacktriangleright g^*(\text{FE}(R)) && \text{by Theorem 2.13} \\
&\equiv Q \blacktriangleright R && \text{by induction hypothesis} \\
&\equiv P.
\end{aligned}$$

Finally, we prove the theorem's statement by making a case distinction on the grammatical category of P . If P is a \top -term, then $\text{FE}(P)$ has only \top -leaves and hence $g(\text{FE}(P)) \equiv g^\top(\text{FE}(P)) \equiv P$, where the first equality is by definition of g and the second was shown above. If P is an F -term, then $\text{FE}(P)$ has only F -leaves and hence $g(\text{FE}(P)) \equiv g^F(\text{FE}(P)) \equiv P$, where the first equality is by definition of g and the second was shown above. If P is a \top - $*$ -term, then it has both \top and

F-leaves and hence, letting $P \equiv Q \bullet \wedge R$,

$$\begin{aligned}
g(\text{FE}(P)) &\equiv g^{\text{T}}(\text{tsd}_1(\text{FE}(P))[\square \mapsto \top]) \bullet \wedge g^*(\text{tsd}_2(\text{FE}(P))) && \text{by definition of } g \\
&\equiv g^{\text{T}}(\text{FE}(Q)) \bullet \wedge g^*(\text{FE}(R)) && \text{by Theorem 2.15} \\
&\equiv Q \bullet \wedge R && \text{as shown above} \\
&\equiv P,
\end{aligned}$$

which completes the proof. □

Proofs for FSCL

B.1 Correctness of f

In order to prove that $f : \text{ST} \rightarrow \text{SNF}$ is indeed a normalization function we need to prove that for all SCL-terms P , $f(P)$ terminates, $f(P) \in \text{SNF}$ and $\text{EqFSCL} \vdash f(P) = P$. To arrive at this result, we prove several intermediate results about the functions f^n and f^c in the order in which their definitions were presented in Section 3.1. For the sake of brevity we will not explicitly prove that these functions terminate. To see that each function terminates consider that a termination proof would closely mimic the proof structure of the lemmas dealing with the grammatical categories of the images of these functions.

Lemma B.1. *For any P^F and P^T , $\text{EqFSCL} \vdash P^F = P^F \triangleleft x$ and $\text{EqFSCL} \vdash P^T = P^T \triangleright x$.*

Proof. We prove both claims simultaneously by induction. In the base case we have $F = F \triangleleft x$ by (SCL7). The base case for the second claim follows from that for the first claim by duality.

For the induction we have $(a \triangleright P^F) \triangleleft Q^F = (a \triangleright P^F) \triangleleft (Q^F \triangleleft x)$ by the induction hypothesis and the result follows from (SCL4). For the second claim we again appeal to duality. \square

Lemma B.2. *The following equations can all be derived by equational logic and EqFSCL.*

1. $(x \triangleright T) \triangleleft \neg y = \neg((x \triangleright T) \triangleleft y)$
2. $(x \triangleleft (y \triangleleft (z \triangleright T))) \triangleright (w \triangleleft (z \triangleright T)) = ((x \triangleleft y) \triangleright w) \triangleleft (z \triangleright T)$
3. $(x \triangleright ((y \triangleright T) \triangleleft (z \triangleleft F))) \triangleleft ((w \triangleright T) \triangleleft (z \triangleleft F)) = ((x \triangleleft (w \triangleright T)) \triangleright (y \triangleright T)) \triangleleft (z \triangleleft F)$
4. $(x \triangleright ((y \triangleright T) \triangleleft (z \triangleleft F))) \triangleleft (w \triangleleft F) = ((\neg x \triangleleft (y \triangleright T)) \triangleright (w \triangleleft F)) \triangleleft (z \triangleleft F)$

Proof. We derive the equations in order.

$$\begin{aligned}
 & (x \triangleright T) \triangleleft \neg y \\
 &= \neg((\neg x \triangleleft F) \triangleright y) && \text{by (SCL1), (SCL2) and (SCL3)} \\
 &= \neg((x \triangleleft F) \triangleright y) && \text{by (SCL8)} \\
 &= \neg((x \triangleright T) \triangleleft y) && \text{by (SCL9)}
 \end{aligned}$$

$$\begin{aligned}
& (x \wp (y \wp (z \vee T))) \vee (w \wp (z \vee T)) \\
&= ((x \wp y) \wp (z \vee T)) \vee (w \wp (z \vee T)) && \text{by (SCL4)} \\
&= ((x \wp y) \vee w) \wp (z \vee T) && \text{by the dual of (SCL10)} \\
& (x \vee ((y \vee T) \wp (z \wp F))) \wp ((w \vee T) \wp (z \wp F)) \\
&= (x \vee ((y \wp F) \vee (z \wp F))) \wp ((w \vee T) \wp (z \wp F)) && \text{by (SCL9)} \\
&= ((x \vee (y \wp F)) \vee (z \wp F)) \wp ((w \vee T) \wp (z \wp F)) && \text{by the dual of (SCL4)} \\
&= (\neg(x \vee (y \wp F)) \vee (w \vee T)) \wp (z \wp F) && \text{by Lemma 3.4 (1)} \\
&= ((\neg x \wp (\neg y \vee T)) \vee (w \vee T)) \wp (z \wp F) && \text{by (SCL1), (SCL2) and (SCL3)} \\
&= ((\neg x \wp (y \vee T)) \vee (w \vee T)) \wp (z \wp F) && \text{by the dual of (SCL8)} \\
&= ((\neg x \wp (y \vee T)) \vee (w \vee (T \vee (y \vee T)))) \wp (z \wp F) && \text{by the dual of (SCL7)} \\
&= ((\neg x \wp (y \vee T)) \vee ((w \vee T) \vee (y \vee T))) \wp (z \wp F) && \text{by the dual of (SCL4)} \\
&= ((x \wp (w \vee T)) \vee (y \vee T)) \wp (z \wp F) && \text{by the dual of Lemma 3.4 (1)} \\
& (x \vee ((y \vee T) \wp (z \wp F))) \wp (w \wp F) \\
&= (\neg x \vee (w \wp F)) \wp (((y \vee T) \wp (z \wp F)) \wp (w \wp F)) && \text{by Lemma 3.4 (1)} \\
&= (\neg x \vee (w \wp F)) \wp ((y \vee T) \wp (z \wp F)) && \text{by (SCL7) and (SCL4)} \\
&= ((\neg x \vee (w \wp F)) \wp (y \vee T)) \wp (z \wp F) && \text{by (SCL4)} \\
&= ((\neg x \wp (y \vee T)) \vee ((w \wp F) \wp (y \vee T))) \wp (z \wp F) && \text{by the dual of (SCL10)} \\
&= ((\neg x \wp (y \vee T)) \vee (w \wp F)) \wp (z \wp F) && \text{by (SCL7) and (SCL4)} \quad \square
\end{aligned}$$

Lemma B.3. *For all $P \in \text{SNF}$, if P is a T -term then $f^n(P)$ is an F -term, if it is an F -term then $f^n(P)$ is a T -term, if it is a T -*-term then so is $f^n(P)$, and*

$$\text{EqFSCL} \vdash f^n(P) = \neg P.$$

Proof. We first prove the claims for T -terms, by induction on P^T . In the base case $f^n(T) = F$. It is immediate that $f^n(T)$ is an F -term. The claim that $\text{EqFSCL} \vdash f^n(T) = \neg T$ is immediate by (SCL1). For the inductive case we have that $f^n((a \wp P^T) \vee Q^T) = (a \vee f^n(Q^T)) \wp f^n(P^T)$, where we assume that $f^n(P^T)$ and $f^n(Q^T)$ are F -terms and that $\text{EqFSCL} \vdash f^n(P^T) = \neg P^T$ and $\text{EqFSCL} \vdash f^n(Q^T) = \neg Q^T$. It follows from the induction hypothesis that $f^n((a \wp P^T) \vee Q^T)$ is an F -term. Furthermore, noting that by the induction hypothesis we may assume that $f^n(P^T)$ and $f^n(Q^T)$ are F -terms, we have:

$$\begin{aligned}
f^n((a \wp P^T) \vee Q^T) &= (a \vee f^n(Q^T)) \wp f^n(P^T) && \text{by definition} \\
&= (a \vee (f^n(Q^T) \wp F)) \wp (f^n(P^T) \wp F) && \text{by Lemma B.1} \\
&= (\neg a \vee (f^n(P^T) \wp F)) \wp (f^n(Q^T) \wp F) && \text{by Lemma 3.4 (2)} \\
&= (\neg a \vee f^n(P^T)) \wp f^n(Q^T) && \text{by Lemma B.1} \\
&= (\neg a \vee \neg P^T) \wp \neg Q^T && \text{by induction hypothesis} \\
&= \neg((a \wp P^T) \vee Q^T). && \text{by (SCL2) and its dual}
\end{aligned}$$

For F -terms we prove our claims by induction on P^F . In the base case $f^n(F) = T$. It is immediate that $f^n(F)$ is a T -term. The claim that $\text{EqFSCL} \vdash f^n(F) = \neg F$ is immediate by the dual of (SCL1). For the inductive case we have that $f^n((a \vee P^F) \wp Q^F) = (a \wp f^n(Q^F)) \vee f^n(P^F)$, where we assume that $f^n(P^F)$ and $f^n(Q^F)$ are T -terms and that $\text{EqFSCL} \vdash f^n(P^F) = \neg P^F$ and

$\text{EqFSCL} \vdash f^n(Q^F) = \neg Q^F$. It follows from the induction hypothesis that $f^n((a \vee P^F) \wedge Q^F)$ is a T -term. Furthermore, noting that by the induction hypothesis we may assume that $f^n(P^F)$ and $f^n(Q^F)$ are T -terms, the proof of derivable equality is dual to that for $f^n((a \wedge P^T) \vee Q^T)$.

To prove the lemma for T -*-terms we first verify that the auxiliary function f_1^n returns a *-term and that for any *-term P , $\text{EqFSCL} \vdash f_1^n(P) = \neg P$. We show this by induction on the number of ℓ -terms in P . For the base cases it is immediate by the above cases for T -terms and F -terms that $f_1^n(P)$ is a *-term. Furthermore, if P is an ℓ -term with a positive determinative atom we have:

$$\begin{aligned}
f_1^n((a \wedge P^T) \vee Q^F) &= (\neg a \wedge f^n(Q^F)) \vee f^n(P^T) && \text{by definition} \\
&= (\neg a \wedge (f^n(Q^F) \vee \mathsf{T})) \vee (f^n(P^T) \wedge \mathsf{F}) && \text{by Lemma B.1} \\
&= (\neg a \vee (f^n(P^T) \wedge \mathsf{F})) \wedge (f^n(Q^F) \vee \mathsf{T}) && \text{by Lemma 3.4 (3)} \\
&= (\neg a \vee f^n(P^T)) \wedge f^n(Q^F) && \text{by Lemma B.1} \\
&= (\neg a \vee \neg P^T) \wedge \neg Q^F && \text{by induction hypothesis} \\
&= \neg((a \wedge P^T) \vee Q^F). && \text{by (SCL2) and its dual}
\end{aligned}$$

If P is an ℓ -term with a negative determinative atom the proof proceeds the same, substituting $\neg a$ for a and applying (SCL3) where needed. For the inductive step we assume that the result holds for all *-terms with fewer ℓ -terms than $P^* \wedge Q^d$ and $P^* \vee Q^c$. We note that each application of f_1^n changes the main connective (not occurring inside an ℓ -term) and hence the result is a *-term. Derivable equality is, given the induction hypothesis, an instance of (the dual of) (SCL2).

With this result we can now see that $f^n(P^T \wedge Q^*)$ is indeed a T -*-term. We note that, by the above, Lemma B.1 implies that $\neg P^T = \neg P^T \wedge \mathsf{F}$. Now we find that:

$$\begin{aligned}
f^n(P^T \wedge Q^*) &= P^T \wedge f_1^n(Q^*) && \text{by definition} \\
&= P^T \wedge \neg Q^* && \text{as shown above} \\
&= (P^T \vee \mathsf{T}) \wedge \neg Q^* && \text{by Lemma B.1} \\
&= \neg((P^T \vee \mathsf{T}) \wedge Q^*) && \text{by Lemma B.2 (1)} \\
&= \neg(P^T \wedge Q^*). && \text{by Lemma B.1}
\end{aligned}$$

Hence for all $P \in \text{SNF}$, $\text{EqFSCL} \vdash f^n(P) = \neg P$. □

Lemma B.4. *For any T -term P and $Q \in \text{SNF}$, $f^c(P, Q)$ has the same grammatical category as Q and*

$$\text{EqFSCL} \vdash f^c(P, Q) = P \wedge Q.$$

Proof. By induction on the complexity of the T -term. In the base case we see that $f^c(\mathsf{T}, P) = P$, which is clearly of the same grammatical category as P . Derivable equality is an instance of (SCL5).

For the inductive step we assume that the result holds for all T -terms of lesser complexity than $a \wedge P^T$. The claim about the grammatical category follows immediately from the induction hypothesis. For the claim about derivable equality we make a case distinction on the grammatical category of the second argument. If the second argument is a T -term, we prove derivable equality

as follows:

$$\begin{aligned}
f^c((a \wp P^\top) \wp Q^\top, R^\top) & \\
&= (a \wp f^c(P^\top, R^\top)) \wp f^c(Q^\top, R^\top) && \text{by definition} \\
&= (a \wp (P^\top \wp R^\top)) \wp (Q^\top \wp R^\top) && \text{by induction hypothesis} \\
&= (a \wp (P^\top \wp (R^\top \wp \top))) \wp (Q^\top \wp (R^\top \wp \top)) && \text{by Lemma B.1} \\
&= ((a \wp P^\top) \wp Q^\top) \wp (R^\top \wp \top) && \text{by Lemma B.2 (2)} \\
&= ((a \wp P^\top) \wp Q^\top) \wp R^\top. && \text{by Lemma B.1}
\end{aligned}$$

If the second argument is an F-term, we prove derivable equality as follows:

$$\begin{aligned}
f^c((a \wp P^\top) \wp Q^\top, R^F) & \\
&= (a \wp f^c(Q^\top, R^F)) \wp f^c(P^\top, R^F) && \text{by definition} \\
&= (a \wp (Q^\top \wp R^F)) \wp (P^\top \wp R^F) && \text{by induction hypothesis} \\
&= (a \wp ((Q^\top \wp \top) \wp (R^F \wp F))) \wp ((P^\top \wp \top) \wp (R^F \wp F)) && \text{by Lemma A.1} \\
&= ((a \wp (P^\top \wp \top)) \wp (Q^\top \wp \top)) \wp (R^F \wp F) && \text{by Lemma B.2 (3)} \\
&= ((a \wp P^\top) \wp Q^\top) \wp R^F. && \text{by Lemma A.1}
\end{aligned}$$

If the second argument is T-*term, the result follows from the case where the second argument is a T-term and (SCL4). \square

Lemma B.5. *For any F-term P and $Q \in \text{SNF}$, $f^c(P, Q)$ is a F-term and*

$$\text{EqFSCL} \vdash f^c(P, Q) = P \wp Q.$$

Proof. The grammatical result is immediate and the claim about derivable equality follows from Lemma B.1, (SCL4) and (SCL7). \square

Lemma B.6. *For any T-*term P and T-term Q , $f^c(P, Q)$ has the same grammatical category as P and*

$$\text{EqFSCL} \vdash f^c(P, Q) = P \wp Q.$$

Proof. By (SCL4) it suffices to prove the claims for f_1^c , i.e., that $f_1^c(P^*, Q^\top)$ is a *-term and that $\text{EqFSCL} \vdash f_1^c(P^*, Q^\top) = P^* \wp Q^\top$. We prove this by induction on the number of ℓ -terms in P^* . In the base case we deal with ℓ -terms and the grammatical claim follows from Lemma B.4. We prove derivable equality as follows, letting $\hat{a} \in \{a, \neg a\}$:

$$\begin{aligned}
f_1^c((\hat{a} \wp P^\top) \wp Q^F, R^\top) &= (\hat{a} \wp f^c(P^\top, R^\top)) \wp Q^F && \text{by definition} \\
&= (\hat{a} \wp (P^\top \wp R^\top)) \wp Q^F && \text{by Lemma B.4} \\
&= ((\hat{a} \wp P^\top) \wp R^\top) \wp Q^F && \text{by (SCL4)} \\
&= ((\hat{a} \wp P^\top) \wp (R^\top \wp \top)) \wp (Q^F \wp F) && \text{by Lemma B.1} \\
&= ((\hat{a} \wp P^\top) \wp (Q^F \wp F)) \wp (R^\top \wp \top) && \text{by Lemma 3.4 (3)} \\
&= ((\hat{a} \wp P^\top) \wp Q^F) \wp R^\top. && \text{by Lemma B.1}
\end{aligned}$$

For the induction step we assume that the result holds for all *-terms with fewer ℓ -terms than $P^* \wp Q^d$ and $P^* \wp Q^c$. In the case of conjunctions the results follow from the induction hypothesis and (SCL4). In the case of disjunctions the results follow immediately from the induction hypothesis, Lemma B.1 and the dual of (SCL10). \square

Lemma B.7. *For any \top -*-term P and \mathbf{F} -term Q , $f^c(P, Q)$ is an \mathbf{F} -term and*

$$\text{EqFSCL} \vdash f^c(P, Q) = P \triangleleft Q.$$

Proof. By Lemma B.4 and (SCL4) it suffices to prove that $f_2^c(P^*, Q^F)$ is an \mathbf{F} -term and that $\text{EqFSCL} \vdash f_2^c(P^*, Q^F) = P^* \triangleleft Q^F$. We prove this by induction on the number of ℓ -terms in P^* . In the base case we deal with ℓ -terms and the grammatical claim follows from Lemma B.4. We derive the remaining claim for ℓ -terms with positive determinative atoms as:

$$\begin{aligned} f_2^c((a \triangleleft P^\top) \vee Q^F, R^F) &= (a \vee Q^F) \triangleleft f^c(P^\top, R^F) && \text{by definition} \\ &= (a \vee Q^F) \triangleleft (P^\top \triangleleft R^F) && \text{by Lemma B.4} \\ &= ((a \vee Q^F) \triangleleft P^\top) \triangleleft R^F && \text{by (SCL4)} \\ &= ((a \vee (Q^F \triangleleft F)) \triangleleft (P^\top \vee \top)) \triangleleft R^F && \text{by Lemma B.1} \\ &= ((a \triangleleft (P^\top \vee \top)) \vee (Q^F \triangleleft F)) \triangleleft R^F && \text{by Lemma 3.4 (3)} \\ &= ((a \triangleleft P^\top) \vee Q^F) \triangleleft R^F. && \text{by Lemma B.1} \end{aligned}$$

For ℓ -terms with negative determinative atoms we derive:

$$\begin{aligned} f_2^c((\neg a \triangleleft P^\top) \vee Q^F, R^F) &= (a \vee f^c(P^\top, R^F)) \triangleleft Q^F && \text{by definition} \\ &= (a \vee (P^\top \triangleleft R^F)) \triangleleft Q^F && \text{by induction hypothesis} \\ &= (a \vee ((P^\top \vee \top) \triangleleft (R^F \triangleleft F))) \triangleleft (Q^F \triangleleft F) && \text{by Lemma B.1} \\ &= ((\neg a \triangleleft (P^\top \vee \top)) \vee (Q^F \triangleleft F)) \triangleleft (R^F \triangleleft F) && \text{by Lemma B.2 (4)} \\ &= ((\neg a \triangleleft P^\top) \vee Q^F) \triangleleft R^F. && \text{by Lemma B.1} \end{aligned}$$

For the induction step we assume that the result holds for all \ast -terms with fewer ℓ -terms than $P^* \triangleleft Q^d$ and $P^* \vee Q^c$. In the case of conjunctions the results follow from the induction hypothesis and (SCL4). In the case of disjunctions note that by Lemma B.3 and the proof of Lemma B.6, we have that $f^n(f_1^c(P^*, f^n(R^F)))$ is a \ast -terms with same number of ℓ -terms as P^* . The grammatical result follows from this fact and the induction hypothesis. Furthermore, noting that by the same argument $f^n(f_1^c(P^*, f^n(R^F))) = \neg(P^* \triangleleft \neg R^F)$, we derive:

$$\begin{aligned} f_2^c(P^* \vee Q^c, R^F) &= f_2^c(f^n(f_1^c(P^*, f^n(R^F))), f_2^c(Q^c, R^F)) && \text{by definition} \\ &= f^n(f_1^c(P^*, f^n(R^F))) \triangleleft (Q^c \triangleleft R^F) && \text{by induction hypothesis} \\ &= \neg(P^* \triangleleft \neg R^F) \triangleleft (Q^c \triangleleft R^F) && \text{as shown above} \\ &= (\neg P^* \vee R^F) \triangleleft (Q^c \triangleleft R^F) && \text{by (SCL3) and (SCL2)} \\ &= (\neg P^* \vee (R^F \triangleleft F)) \triangleleft (Q^c \triangleleft (R^F \triangleleft F)) && \text{by Lemma B.1} \\ &= (P^* \vee Q^c) \triangleleft (R^F \triangleleft F) && \text{by Lemma 3.4 (1)} \\ &= (P^* \vee Q^c) \triangleleft R^F. && \text{by Lemma B.1} \end{aligned}$$

This completes the proof. \square

Lemma B.8. *For any $P, Q \in \text{SNF}$, $f^c(P, Q)$ is in SNF and*

$$\text{EqFSCL} \vdash f^c(P, Q) = P \triangleleft Q.$$

Proof. By the four preceding lemmas it suffices to show that $f^c(P^\top \triangleleft Q^*, R^\top \triangleleft S^*)$ is in SNF and that $\text{EqFSCL} \vdash f^c(P^\top \triangleleft Q^*, R^\top \triangleleft S^*) = (P^\top \triangleleft Q^*) \triangleleft (R^\top \triangleleft S^*)$. By (SCL4), in turn, it suffices to prove that $f_3^c(P^*, Q^\top \triangleleft R^*)$ is a $*$ -term and that $\text{EqFSCL} \vdash f_3^c(P^*, Q^\top \triangleleft R^*) = P^* \triangleleft (Q^\top \triangleleft R^*)$. We prove this by induction on the number of ℓ -terms in R^* . In the base case we have that $f_3^c(P^*, Q^\top \triangleleft R^\ell) = f_1^c(P^*, Q^\top) \triangleleft R^\ell$. The results follow from Lemma B.6 and (SCL4).

For conjunctions the result follows from the induction hypothesis and (SCL4) and for disjunctions it follows from Lemma B.6 and (SCL4). \square

Theorem 3.7. *For any $P \in \text{ST}$, $f(P)$ terminates, $f(P) \in \text{SNF}$ and $\text{EqFSCL} \vdash f(P) = P$.*

Proof. By induction on the complexity of P . If P is an atom, the result is by (SCL5), (SCL6) and its dual. If P is \top or F the result is by identity. For the induction we get the result by Lemma B.3, Lemma B.8 and (SCL2). \square

B.2 Correctness of g

Theorem 3.17. *For all $P \in \text{SNF}$, $g(\text{SE}(P)) \equiv P$.*

Proof. We first prove that for all \top -terms P , $g^\top(\text{SE}(P)) \equiv P$, by induction on P . In the base case $P \equiv \top$ and we have $g^\top(\text{SE}(P)) \equiv g^\top(\top) \equiv \top \equiv P$. For the inductive case we have $P \equiv (a \triangleleft Q^\top) \circledast R^\top$ and

$$\begin{aligned} g^\top(\text{SE}(P)) &\equiv g^\top(\text{SE}(Q^\top) \triangleleft a \triangleright \text{SE}(R^\top)) && \text{by definition of SE} \\ &\equiv (a \triangleleft g^\top(\text{SE}(Q^\top))) \circledast g^\top(\text{SE}(R^\top)) && \text{by definition of } g^\top \\ &\equiv (a \triangleleft Q^\top) \circledast R^\top && \text{by induction hypothesis} \\ &\equiv P. \end{aligned}$$

Similarly we see that for all F -terms P , $g^\text{F}(\text{SE}(P)) \equiv P$, by induction on P . In the base case $P \equiv \text{F}$ and we have $g^\text{F}(\text{SE}(P)) \equiv g^\text{F}(\text{F}) \equiv \text{F} \equiv P$. For the inductive case we have $P \equiv (a \circledast Q^\text{F}) \triangleleft R^\text{F}$ and

$$\begin{aligned} g^\text{F}(\text{SE}(P)) &\equiv g^\text{F}(\text{SE}(R^\text{F}) \triangleleft a \triangleright \text{SE}(Q^\text{F})) && \text{by definition of SE} \\ &\equiv (a \circledast g^\text{F}(\text{SE}(Q^\text{F}))) \triangleleft g^\text{F}(\text{SE}(R^\text{F})) && \text{by definition of } g^\text{F} \\ &\equiv (a \circledast Q^\text{F}) \triangleleft R^\text{F} && \text{by induction hypothesis} \\ &\equiv P. \end{aligned}$$

Now we check that for all ℓ -terms P , $g^\ell(\text{SE}(P)) \equiv P$. We observe that either $P \equiv (a \triangleleft Q^\top) \circledast R^\text{F}$ or $P \equiv (\neg a \triangleleft Q^\top) \circledast R^\text{F}$. In the first case we have

$$\begin{aligned} g^\ell(\text{SE}(P)) &\equiv g^\ell(\text{SE}(Q^\top) \triangleleft a \triangleright \text{SE}(R^\text{F})) && \text{by definition of SE} \\ &\equiv (a \triangleleft g^\top(\text{SE}(Q^\top))) \circledast g^\text{F}(\text{SE}(R^\text{F})) && \text{by definition of } g^\ell \\ &\equiv (a \triangleleft Q^\top) \circledast R^\text{F} && \text{as shown above} \\ &\equiv P. \end{aligned}$$

In the second case we have that

$$\begin{aligned} g^\ell(\text{SE}(P)) &\equiv g^\ell(\text{SE}(R^\text{F}) \triangleleft \neg a \triangleright \text{SE}(Q^\top)) && \text{by definition of SE} \\ &\equiv (\neg a \triangleleft g^\top(\text{SE}(Q^\top))) \circledast g^\text{F}(\text{SE}(R^\text{F})) && \text{by definition of } g^\ell \\ &\equiv (\neg a \triangleleft Q^\top) \circledast R^\text{F} && \text{as shown above} \\ &\equiv P. \end{aligned}$$

We now prove that for all $*$ -terms P , $g^*(\text{SE}(P)) \equiv P$, by induction on P modulo the complexity of ℓ -terms. In the base case we are dealing with ℓ -terms. Because an ℓ -term has neither a cd nor a dd we have $g^*(\text{SE}(P)) \equiv g^\ell(\text{SE}(P)) \equiv P$, where the first equality is by definition of g^* and the second was shown above. For the induction we have either $P \equiv Q \circledast R$ or $P \equiv Q \circledvee R$. In the first case note that by Theorem 3.13, $\text{SE}(P)$ has a cd and no dd. So we have

$$\begin{aligned} g^*(\text{SE}(P)) &\equiv g^*(\text{cd}_1(\text{SE}(P))[\square \mapsto \top]) \circledast g^*(\text{cd}_2(\text{SE}(P))) && \text{by definition of } g^* \\ &\equiv g^*(\text{SE}(Q)) \circledast g^*(\text{SE}(R)) && \text{by Theorem 3.13} \\ &\equiv Q \circledast R && \text{by induction hypothesis} \\ &\equiv P. \end{aligned}$$

In the second case, again by Theorem 3.13, P has a dd and no cd. So we have that

$$\begin{aligned} g^*(\text{SE}(P)) &\equiv g^*(\text{dd}_1(\text{SE}(P))[\square \mapsto \text{F}]) \circledvee g^*(\text{dd}_2(\text{SE}(P))) && \text{by definition of } g^* \\ &\equiv g^*(\text{SE}(Q)) \circledvee g^*(\text{SE}(R)) && \text{by Theorem 3.13} \\ &\equiv Q \circledvee R && \text{by induction hypothesis} \\ &\equiv P. \end{aligned}$$

Finally, we prove the theorem's statement by making a case distinction on the grammatical category of P . If P is a \top -term, then $\text{SE}(P)$ has only \top -leaves and hence $g(\text{SE}(P)) \equiv g^\top(\text{SE}(P)) \equiv P$, where the first equality is by definition of g and the second was shown above. If P is a F -term, then $\text{SE}(P)$ has only F -leaves and hence $g(\text{SE}(P)) \equiv g^\text{F}(\text{SE}(P)) \equiv P$, where the first equality is by definition of g and the second was shown above. If P is a \top - $*$ -term, then it has both \top and F -leaves and hence, letting $P \equiv Q \circledast R$,

$$\begin{aligned} g(\text{SE}(P)) &\equiv g^\top(\text{tsd}_1(\text{SE}(P))[\square \mapsto \top]) \circledast g^*(\text{tsd}_2(\text{SE}(P))) && \text{by definition of } g \\ &\equiv g^\top(\text{SE}(Q)) \circledast g^*(\text{SE}(R)) && \text{by Theorem 3.16} \\ &\equiv Q \circledast R && \text{as shown above} \\ &\equiv P, \end{aligned}$$

which completes the proof. □

Bibliography

- [AGH06] K. Arnold, J. Gosling, and D.C. Holmes, *The Java Programming Language*, The Java Series, Addison-Wesley, 2006.
- [Blo11] A. Blok, *Side-Effecting Logic*, 2011, unpublished.
- [BP10a] J.A. Bergstra and A. Ponse, *On Hoare-McCarthy Algebras*, 2010, available as arXiv:1012.5059v1 [cs.LO].
- [BP10b] ———, *Short-Circuit Logic*, 2010, available as arXiv:1010.3674v3 [cs.LO].
- [BP11] ———, *Proposition Algebra*, ACM Transactions on Computational Logic **12** (2011), 21:1–21:36.
- [BP12] ———, *Proposition Algebra and Short-Circuit Logic*, Fundamentals of Software Engineering (F. Arbab and M. Sirjani, eds.), Lecture Notes in Computer Science, vol. 7141, Springer Berlin / Heidelberg, 2012, pp. 15–31.
- [BW96] P.E. Black and P.J. Windley, *Inference Rules for Programming Languages with Side Effects in Expressions*, In International Conference on Theorem Proving in Higher Order Logics, Springer, 1996, pp. 51–60.
- [GPFW96] J. Gu, P.W. Purdom, J. Franco, and B.W. Wah, *Algorithms for the Satisfiability (SAT) Problem: A Survey*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, 1996, pp. 19–152.
- [Hoa85] C.A.R. Hoare, *A Couple of Novelties in the Propositional Calculus*, Zeitschrift für Mathematische Logik und Grundlagen der Mathematik **31** (1985), no. 9-12, 173–178.
- [Nor97] M. Norrish, *An abstract dynamic semantics for C*, Tech. Report 421, Computer Laboratory, University of Cambridge, 1997.
- [Pon11] A. Ponse, *Truth Table Semantics for Short-Circuit Logic and for Proposition Algebra*, 2011, unpublished.
- [PVdZ06] A. Ponse and M. Van der Zwaag, *An Introduction to Program and Thread Algebra*, Logical Approaches to Computational Barriers (A. Beckmann, U. Berger, B. Löwe, and J. Tucker, eds.), Lecture Notes in Computer Science, vol. 3988, Springer Berlin / Heidelberg, 2006, pp. 445–458.
- [Reg10] B.C. Regenboog, *Reactive Valuations*, 2010, Master of Logic thesis, University of Amsterdam, available as arXiv:1101.3132v1 [cs.LO].